Honeywell

2

Rapid Prototyping of Application Specific Signal Processors (RASSP) Program – Study Phase

Final Report
For Period of May 12, 1992 through October 12, 1992

A Defense Advanced Research Projects Agency (DARPA) Program

Contract MDA972-92-C-0057
ARPA Order No. 9219/1

12 October 1992

Submitted to:

Defense Advanced Research Projects Agency
Electronic Systems Technology Office
3701 North Fairfax Drive
Arlington, VA 22203-1714

Systems and Research Center
10701 Lyndale Avenue South
Bloomington, Minnesota 55420

Rapid Prototyping of Application Specific Signal Processors (RASSP) Program – Study Phase

Final Report
For Period of May 12, 1992 through October 12, 1992

A Defense Advanced Research Projects Agency (DARPA) Program

Contract MDA972-92-C-0057
ARPA Order No. 9219/1

12 October 1992

Submitted to:

Defense Advanced Research Projects Agency
Electronic Systems Technology Office
3701 North Fairfax Drive
Arlington, VA 22203-1714

Systems and Research Center
10701 Lyndale Avenue South
Bloomington, Minnesota 55420

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 12 October 1992	3. REPORT TYPE AND DATES COVERED Final Report for Period 5/12/92 - 10/12/92

4. TITLE AND SUBTITLE
Rapid Prototyping of Application Specific Signal Processors (RASSP) Program - Study Phase

5. FUNDING NUMBERS

6. AUTHOR(S)
Fred Malver, Andrzej Peczalski, Wing Au, Jonathon Krueger, David Lee

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Honeywell Inc.
Systems and Research Center
10701 Lyndale Avenue South
Bloomington, MN 55420

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
Electronic Systems Technology Office
3701 North Fairfax Drive
Arlington, VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
MDA972-92-C-0057

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unrestricted

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This is the final report for the Rapid Prototyping of Applications Specific Signal Processors (RASSP) Program - Study Phase. This study represents a five-month contract effort, DARPA contract number MDA972-92-C-0057, performed by Honeywell's Systems and Research Center for the Defense Advanced Research Projects Agency, DARPA.

The broad objective of this study was to produce information that would aid the Government program manager to manage the risks inherent in the implementation phase of the RASSP program. This meant (1) identifying the risks and problem areas that might be encountered during the implementation phase and (2) suggesting risk reducers or solutions that could be used to minimize or solve these problems. Ultimately this meant recommending an approach for the implementation phase and also identifying potential areas for further work.

The RASSP program embraces two tightly coupled focuses; one related to process or methodology, and the other related to target prototype or product systems. These are discussed in detail in this final report.

14. SUBJECT TERMS
Rapid Prototyping, Application Specific Signal Processors, Design, Manufacturing and Field Support Process and Methodology Infrastructure, Model Year Upgrades, Target Applications - ATR, EW, Communications, and SIGINT.

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Same as report

Table of Contents

Table of Contents

Table of Contents

List of Tables

List of Figures

List of Figures

List of Figures

Section 1
Study Overview and Summary

This is the final report for the Rapid Prototyping of Application Specific Signal Processors (RASSP) Program - Study Phase. This study represents a five-month contract effort, DARPA contract number MDA972-92-C-0057, performed by Honeywell's Systems and Research Center for the Defense Advanced Research Projects Agency, DARPA.

1.1 Study Objective

The broad objective of this study was to produce information that would aid the Government program manager to manage the risks inherent in the implementation phase of the RASSP program. This meant (1) identifying the risks and problem areas that might be encountered during the implementation phase, and (2) suggesting risk reducers or solutions that could be used to minimize or solve these problems. Ultimately this meant recommending an approach for the implementation phase and also identifying potential areas for further work.

Our proposal stated the intention to answer the question, "where are the risks?" This final report will document our findings.

1.2 Study Approach and General Methodology

The RASSP program embraces two tightly coupled focuses; one related to process or methodology, and the other related to target prototype or product systems. Key capabilities associated with the rapid prototyping process are listed as columns of the matrix in Figure 1.2-1. Key program goals associated with the target systems were the proposed generic signal processor architecture (shown in Figure 1.2-1) and the model year concept. Suggested target systems applications were automatic target acquisition, tracking, and recognition (ATR), electronic countermeasurers (EC), communications (COMM), and signal intelligence (SIGINT).

Our study approach is summarized by the assessment matrix shown in Figure 1.2-1. We examined all of the study phase considerations (cited in the RASSP RFP and indicated as columns across the matrix) over the phases of prototype or product development (listed as the rows of the assessment matrix). The prototype or product development phases covered the design/development process from application specific concurrent engineering planning (ASCEP) to field insertion.

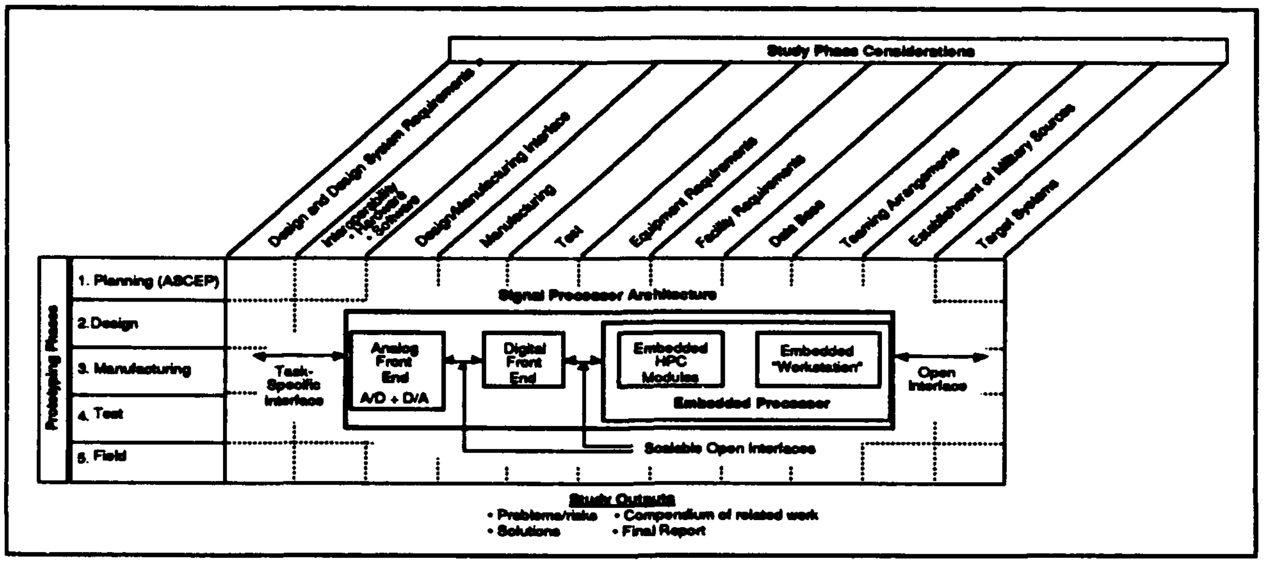


Figure 1.2-1. Study Assessment Matrix

The study methodology that evolved from this approach is shown in Figure 1.2-2. We postulated a conceptual RASSP ASES "factory" and the top level functions that would be performed by this factory. These functions appear in Figure 1.2-2 as an integrated concurrent process flow for RASSP. They also were used as task elements for the study. Each element in this set of integrated parallel flows was examined in terms of its inputs and outputs and the function performed. The risk assessment was based upon the state of the art in each of these areas by examining ongoing activities, standards, issues, and making recommendations. This approach was straightforward and thorough.

To test this approach and confirm our results, we further postulated a test case based upon a conceptualized Touchstone processor as the candidate embedded signal processor.

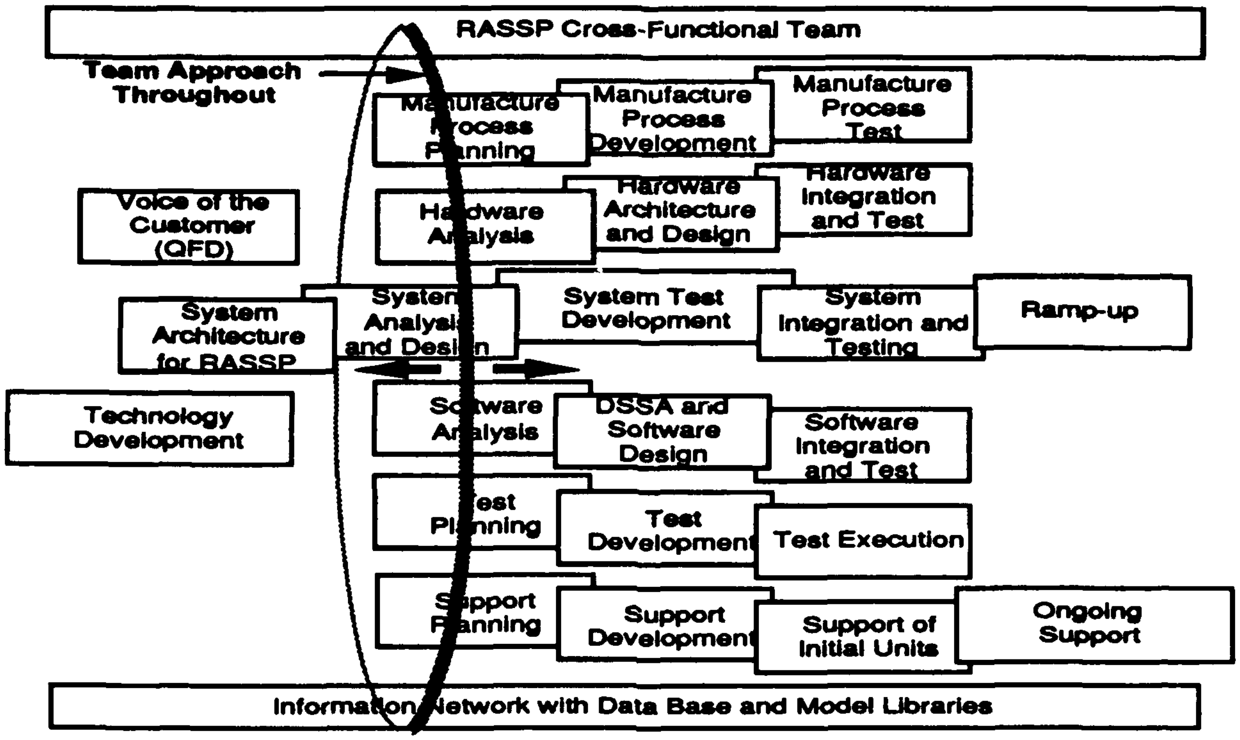

RASSP Cross-Functional Team
Team Approach Throughout
Manufacture Process Planning
Manufacture Process Development
Manufacture Process Test
Voice of the Customer (QFD)
Hardware Analysis
Hardware Architecture and Design
Hardware Integration and Test
System Architecture for RASSP
System Analysis and Design
System Test Development
System Integration and Testing
Ramp-up
Technology Development
Software Analysis
DSSA and Software Design
Software Integration and Test
Test Planning
Test Development
Test Execution
Support Planning
Support Development
Support of Initial Units
Ongoing Support
Information Network with Data Base and Model Libraries

Figure 1.2-2. Study Methodology and Process Flow for RASSP

1.3 General Study Technical Results

Stated in broadest generalizations regarding product, process, and culture, the basic study results can be summarized as follows:

- RASSP Product—The level of risk associated with RASSP prototype or product development will be directly related to how aggressively the application (we have recommended ATR) and its implementing technologies are pushed. This should be controllable by the government's RASSP program manager. The generic signal processor architecture makes sound technical sense and will be challenged to extend the embedded general-purpose processor's interface as far forward as possible. The model year concept makes sound business sense but may require changes to traditional DoD procurement practices. The Touchstone multi-processor architecture (see attached Honeywell proprietary addendum) can provide the flexibility and upgradability sought for RASSP.

- RASSP Process—No insurmountable technical barriers (i.e., no "show stoppers") were found preventing use of the proposed generic signal processor architecture, prohibiting implementation of the model year concept, nor blocking the implementation of a highly effective and efficient, seamlessly integrated concurrent RASSP methodology. Our assumption of a RASSP factory did identify many technical challenges; but with available workarounds and evolving solutions, these presented a manageable and acceptable level of technical risk. The model year concept directly influences the product development process at key points and in ways that can enhance this process. These key points in the process are identified and examined in detail. The stage that we have called "high-level description capture" (see Figure 1.3-2) and its associated hardware and software modeling is a critical juncture in the design process where important hardware and software implementation decisions must be made. This key step and the steps leading to it need improved tools and integration.

- RASSP Culture—Organizational and business barriers to RASSP implementation will be the most difficult to overcome. Institutional barriers mostly created by Federal Acquisition Regulations (FARs) and Cost Accounting Standards (CASs) can prevent the RASSP program from fully exploiting the benefits of high-volume commercial manufacturing.

We have enclosed a videotape entitled, "Mod IV World Class Circuit Board Line," to emphasize the point relating to FARs/CASs and to demonstrate the manufacturing automation available for improving affordability. This video was made at Honeywell's Home and Building Control Division (HBCD) in Golden Valley, Minnesota. It shows a state-of-the-art automated PCB assembly line used in the manufacture of damper motors. This line can produce two SMT or through-hole boards per minute while simultaneously supporting the manufacture of single unit lot sizes. The high, automated throughput is necessary for producing more-affordable RASSP systems. Also, as military systems employ finer and finer pitch parts, human handling of these parts is increasingly prone to causing parts failures. Therefore, human handling of parts should increasingly be avoided. Thus the need for more automated PCB assembly in producing advanced technology military systems is viewed as an eventual necessity as well as a cost and risk reducer. Unfortunately, the relatively low volumes associated with military electronics products as compared to commercial electronics make it difficult to justify the initial costs of setting up such automated production for military use only. We would like to see RASSP PCB assembly performed on a highly automated, high volume commercial assembly line; for example, a next generation assembly line following after the one shown in the accompanying HBCD video. This simply may not be possible because of the aforementioned government procurement regulations and commercial industry's inability to accept them.

Our vision of a RASSP factory for producing application specific electronic systems (ASES) is an outgrowth of best commercial practices and ongoing DARPA-sponsored ASIC, ASEM, and infrastructure programs. This concept is diagrammed in Figure 1.3-1. It is characterized by an integrated and sustaining

- Engineering department;
- Flexible, open manufacturing facility (preferably commercial);
- Product support organization;
- Community of military and commercial vendors of hardware and software products;
- "Global" interconnecting network and support infrastructure.

all working together for the rapid acquisition and model year upgrading of application specific electronic systems.

The RASSP/ASES factory should be capable of manufacturing various existing configurations of an ASES within days, manufacturing upgraded (40% performance) configurations of an ASES within one to two years, and sustaining a development pipeline for model year upgrades. This factory would provide direct access for customers through standard network and data exchange formats including access to a library of products, controlled access to proprietary hardware and software design tools, and access to testing requirements and potentially even test equipment. The factory facility would be modular, scaleable, highly flexible, and highly automated. It would be supported by an educated and specially trained, highly agile workforce. While the vision of such a factory may currently seem fanciful, industry is moving in these directions. The RASSP program has the potential to accelerate this progress with little inherent risk but huge potential benefit to the military.

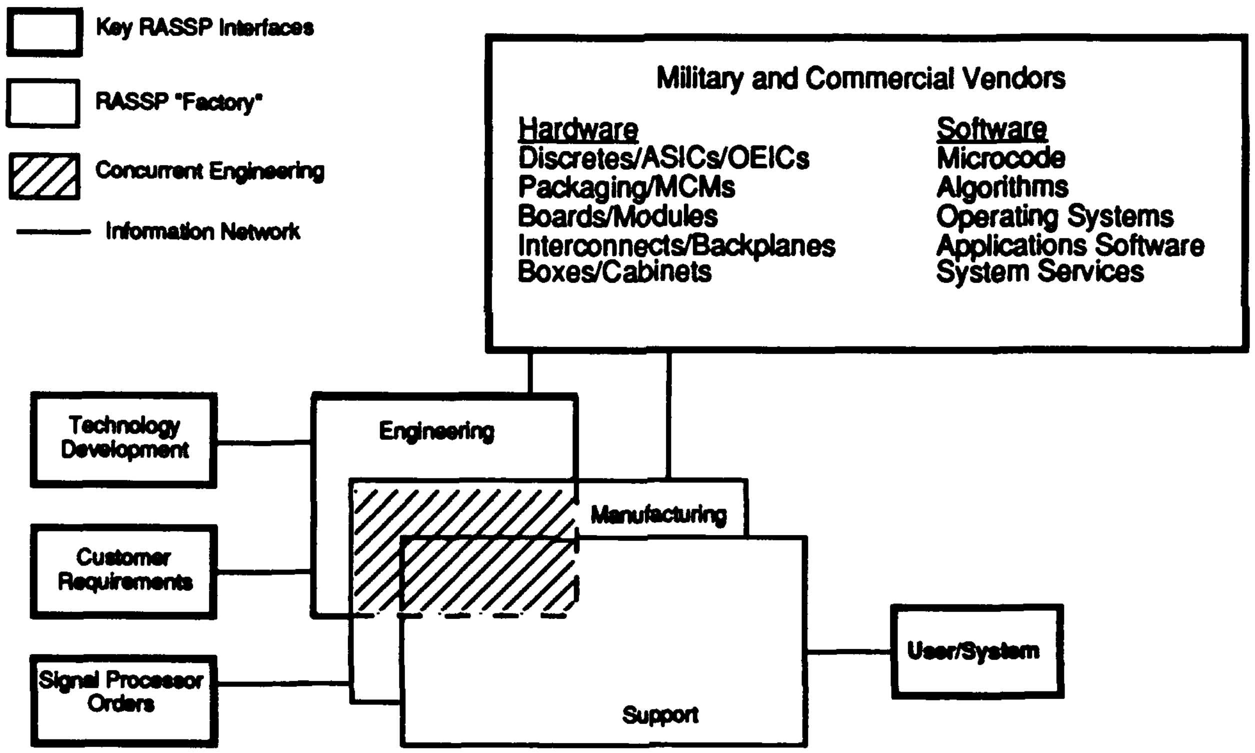


Figure 1.3-1. Key Elements of a RASSP Factory

The following three subsections will more extensively summarize our risk assessment findings in the areas of product, process, and cultural issues.

1.3.1 RASSP Product

Honeywell evaluated three target systems for RASSP application: an automatic target recognition (ATR) system, an electronic warfare (EW) system and a communications system. Six tradeoff areas were use in this evaluation: (1) U. S. Critical Need, (2) System Considerations,

(3) Technology Maturity, (4) Manufacture, Production, and Field, (5) Model-Year Upgradability, and (6) Logistics. Criteria were defined in each area and each target system was evaluated and scored against these criteria. The ATR system scored highest, 157 points, the comm system was second with 140 points, and EW system was third with 135 points.

State-of-the-art ATR has demonstrated success in automatic target cueing, aiding an operator to locate targets. Automatic target recognition, however, has had only limited success under mission conditions although the technology is improving. Based on our understanding of the SOA of ATR technology, five deterrents have been identified that could jeopardize the success of an ATR RASSP demonstration: (1) limited successful operational scenarios, (2) no common acceptable standards, (3) unavailability of high-performance military qualified processors, (4) difficult and high cost performance evaluation, and (5) lack of sophisticated real-time algorithms. The RASSP program must directly address the first three during the implementation phase. RASSP only needs to indirectly support the other two.

One of the strengths of an ATR system is that it has numerous model year upgrade options. The baseline is an automatic target cuer (ATC) system; the first upgrade to a multi-platform ATR expands the user community; the second upgrade to precision strike capability incorporates multiple sensor and multiple mission ATR technologies. The third upgrade increases the ATR's functionality to reconnaissance and surveillance applications.

ATR is of interest to many users. A clear development path with many ATR applications is projected. ATR technology is expected to mature and flourish by the end of this decade.

1.3.2 RASSP Methodology

Of the two primary facets of the RASSP program, product and process, process (i.e., rapid prototyping methodology) in our view is the more significant of the two. New processors can be developed in a business as usual manner, but it is the RASSP process that ultimately will make a real difference. The challenge in RASSP, however, is not so much in process development as it is in process integration (i.e., the integration of many pieces that either exist or are in some state of development). Since all the pieces for this process are not currently available, a structure or architecture for the process must be defined as a framework into which these pieces will ultimately fit. This structure, with its information infrastructure and data base, is a superset of the CAD framework concept or perhaps a multi-dimensional expansion of it into additional application-specific product development domains.

So where are the risks in trying to integrate such a process? The basic risk is that it cannot be accomplished, at least not in the limited time span of the RASSP implementation phase; i.e., the integration of this product development process may be a slowly evolving process where the end is never reached. The risk reducer here is to establish a shared vision of what this process should be. Then we would at least all be working toward the same goal. The RASSP implementation phase has the potential at a minimum (also at minimum risk) of defining this goal and more optimistically of moving us well along the path to reaching this goal, creating a momentum for the process that ultimately will be maintained by market forces. Driven by the dynamics of supply and demand, the process will evolve and optimize its effectiveness.

The process for rapid prototyping and development of application specific electronic systems (ASES) will be built upon past and ongoing DARPA and industry programs (See Figure 1.3.2-1.) This process, as with the product, must be designed to accommodate upgrades; it must evolve as the product and process technologies evolve.

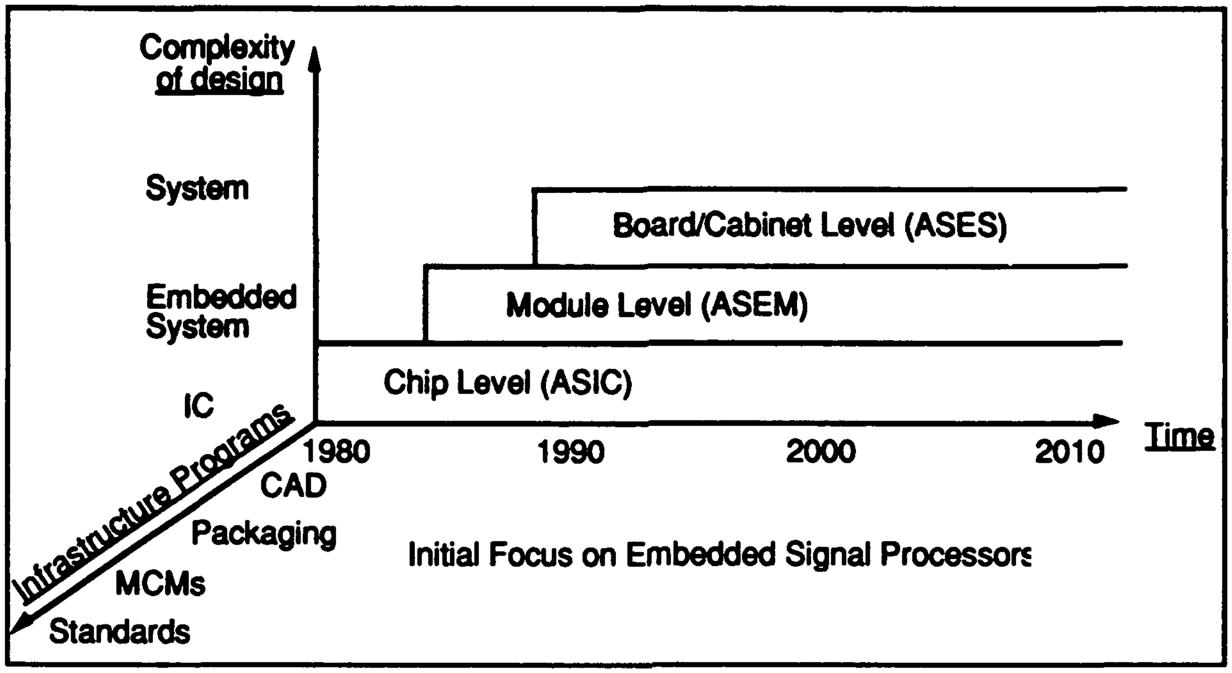
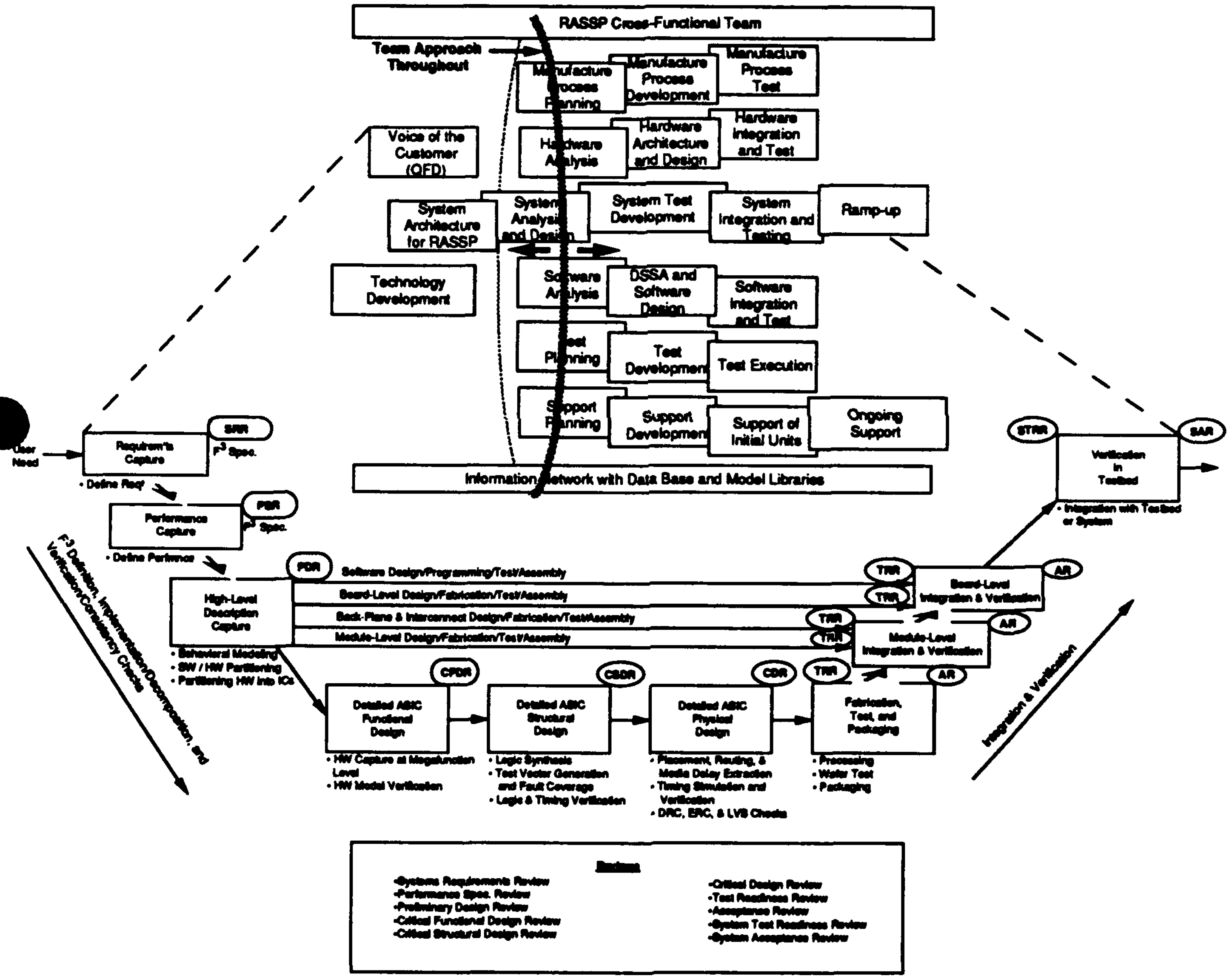


Figure 1.3.2-1. Past and Ongoing DARPA Programs Provide the Foundation for RASSP

The description of a highly structured concurrent product development environment/process with an associated (as seamless as possible) design methodology is a central part of this final report. Figure 1.3.2-2 attempts to diagram this overall process. This process with its concurrency can establish the environment for rapid and seamless development of a product, including both hardware and software, from concept to fieldable prototype and/or fully qualified product and to support that product's model year updates. This assertion infers the concept of a very structured, highly integrated process for product definition and development together with the support of an integrally networked infrastructure.

Although such a process does not exist today, we have a good sense of what it should be. Some elements of this process, such as ASIC design methodology, do exist. Other elements, such as ASEM/MCM design methodology, are being developed under other programs. Elements of system design, software design, testing, and information infrastructure need to be developed by RASSP. The overall "big picture" process is continuously developing and evolving. To tie all these elements together, a system design formalism is necessary. This formalism, described in this report, will provide the consistency of product and process descriptions necessary to facilitate the overall concurrent product development process. More importantly, **this formalism greatly facilitates a product's model year upgrading and the likelihood of achieving interoperability within both the product and process domains.**

Concurrent Product Development Environment with Seamless Design Methodology



RASSP Cross-Functional Team
Team Approach Throughout
Manufacture Process Planning
Manufacture Process Development
Manufacture Process Test
Voice of the Customer (QFD)
Hardware Analysis
Hardware Architecture and Design
Hardware Integration and Test
System Architecture for RASSP
System Analysis and Design
System Test Development
System Integration and Testing
Ramp-up
Technology Development
Software Analysis
DSSA and Software Design
Software Integration and Test
Test Planning
Test Development
Test Execution
Support Planning
Support Development
Support of Initial Units
Ongoing Support
Information Network with Data Base and Model Libraries
User Need
Requirem'ts Capture
SRR
F^3 Spec.
• Define Req'
Performance Capture
PSR
F^3 Spec.
• Define Perfrmnce
F^3 Definition, Implementation/Decomposition, and Verification/Consistency Checks
High-Level Description Capture
PDR
• Behavioral Modeling
• SW / HW Partitioning
• Partitioning HW into ICs
Software Design/Programming/Test/Assembly
Board-Level Design/Fabrication/Test/Assembly
Back-Plane & Interconnect Design/Fabrication/Test/Assembly
Module-Level Design/Fabrication/Test/Assembly
CFDR
Detailed ASIC Functional Design
• HW Capture at Megafunction Level
• HW Model Verification
CSDR
Detailed ASIC Structural Design
• Logic Synthesis
• Test Vector Generation and Fault Coverage
• Logic & Timing Verification
CDR
Detailed ASIC Physical Design
• Placement, Routing, & Media Delay Extraction
• Timing Simulation and Verification
• DRC, ERC, & LVS Checks
TRR
Fabrication, Test, and Packaging
AR
• Processing
• Wafer Test
• Packaging
TRR
Module-Level Integration & Verification
AR
TRR
Board-Level Integration & Verification
AR
Integration & Verification
STRR
Verification in Testbed
SAR
• Integration with Testbed or System
Reviews
•Systems Requirements Review
•Performance Spec. Review
•Preliminary Design Review
•Critical Functional Design Review
•Critical Structural Design Review
•Critical Design Review
•Test Readiness Review
•Acceptance Review
•System Test Readiness Review
•System Acceptance Review

Figure 1.3.2-2. Concurrent Product Development Environment/Process for ASES

Why is this system development process so important and how does it relate to RASSP implementation risks? (1) At the point in the hierarchical system design process where software and hardware implementation tradeoffs are made and system SW/HW partitioning is performed, there does not seem to be a well-defined common set of metrics that anchor this decision point; and subsequently these two processes (of software and hardware development) seem to be performed orthogonally, not linked in a manner that would help reduce design risk and improve design tracking. (2) The higher levels of concurrent product development methodology are still fairly immature and new tools are just emerging. Thus it is helpful if there is a broader context to identify where new tools are needed and to tie these new tools and procedures together, aiming at the seamless product development environment envisioned for RASSP. (3) **Most importantly, this formalism is itself a risk reducer, not just for the RASSP implementation phase, but for product development and product upgrading in general.**

The design process formalism that will be described is built upon the three basic steps of the system engineering process: definition, implementation, and verification. The design itself, both software or hardware, must be described in a canonical form that can provide a measure of the design's completeness. The design attributes of form, fit, and function (F^3) will be used as the specifications necessary to completely define a design. This is nothing new. The airline industry has been using F^3 practices for years, and the DoD has occasionally applied it to new systems. At any and all levels of a hierarchical design, each element must have form, fit, and functional descriptions for that element, whether hardware or software, to be completely defined.

When viewed in the larger context of concurrent product development, the attributes of form, fit, and function provide a concise basis for tieing the complete product description together including introducing the "ilities" into the design process right up front in the requirements capture and performance capture phases of the process. As the design is subsequently defined and hierarchically decomposed, consideration of the "ilities" on the design can be imposed and consistently traced throughout the design and configuration management processes. Design decisions thus include consideration of manufacturability, quality, testability, reliability, affordability, etc., as an integral part of the design process. Again this can apply to both hardware and software.

The F^3 process sounds naively simple, but it is very powerful. It is a mechanism, still utilized by the airline industry, for achieving interoperability and upgradable systems. Its formalism can help bind the RASSP process together.

1.3.3 Cultural Issues

1.3.3.1 Military versus Commercial Business Practices—The RASSP program provides an opportunity to directly address issues associated with military acceptance of commercial business practices, data rights, and commercial parts. These long-standing issues have frustrated the military and commercial industrial sectors for years. While various military studies and government commissions have recommended the adoption of best commercial practices, and procurement reform initiatives are being pursued, federal procurement and contract audit practices have created policies that still force a separation between these two domains. These policies and practices include:

- Strict use of Federal Acquisition Regulations (FAR) and Cost Accounting Standards (CAS)
- Right of government agencies to audit books of those companies bidding contracts
- The restriction of profit and fee companies are allowed to bid
- The forfeiture of proprietary and company secret information
- Military specifications and standards
- International Traffic and Arms Regulations (ITAR) restrictions.

A vehicle that partially overcomes these barriers is the Cooperative Research and Development Agreement (CRADA). However, CRADA does not cover the situation when the government provides funding directly to a commercial company. CRADA has been around for just over a year and is not yet an accepted way of doing business. Today commercial companies typically establish separate "federal" divisions for doing government business.

This situation is particularly frustrating to a program such as RASSP where high-volume, highly automated, advanced commercial manufacturing facilities may not be accessible by this program. If the RASSP implementation phase could be contracted in such a way as to waive restrictive FARs and CASs, then a company might be willing to participate in an experiment that would attempt to combine low-volume military production into a high-volume commercial line. If this cannot be accomplished, then we would recommend setting up a separate advanced assembly line for RASSP, apart from but in parallel with a similar commercial line. The synergism between these two lines would still be beneficial to both and a risk reducer to the RASSP program. Our long-range hope is that the barriers between military and commercial businesses can be overcome or eliminated.

1.3.3.2 Concurrent Product Development—The product development environment recommended for successful RASSP implementation is a concurrent or integrated product development environment. This environment is viewed as a significant risk reducer. Fortunately, industry is moving in the direction toward more empowerment of the workforce, more participation in problem solving, and more ownership in product improvement. This is a cultural issue that requires the commitment of a company's management. The RASSP program manager should be aware of such a company commitment.

1.3.3.3 Automated Manufacturing—Automated assembly is an affordability driver and risk reducer for the RASSP implementation phase. The automation of circuit board assembly is taking place in the electronics industry as illustrated by the attached HBCD videotape. Cultural issues associated with labor grade, education, and training must be addressed. The skills required to keep robotic machines working and computer-aided manufacturing and testing operations functioning necessitate upgrading the traditional manufacturing work force. The success of the RASSP implementation phase will be jeopardized if automated manufacturing and an adequately trained work force are not accessible.

1.3.3.4 Satisfying Mil-Spec Requirements—It typically can take months of testing to fully qualify a product for meeting military standards. The time required for qualification and reliability testing will reduce the rapidity of prototype development. A proposed solution is diagrammed in Figure 1.3.3.4-1.

Reliance on qualified manufacturers and qualified parts can be employed to produce a fieldable prototype. While the prototype is being evaluated in the field, system-level or assembly-level qualification and reliability testing can be performed in parallel. Also in parallel, work can begin on the next model year upgrade. Paralleling product development activities help compress the prototype and product development cycles.

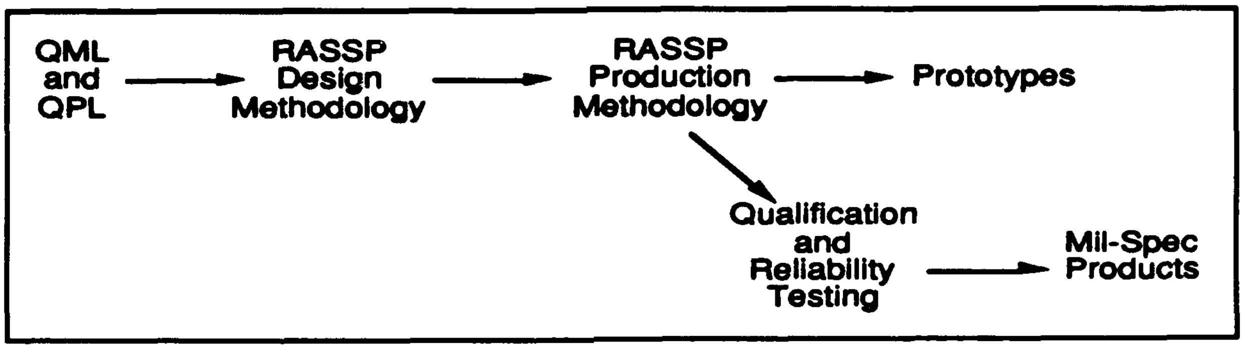

QML and QPL
RASSP Design Methodology
RASSP Production Methodology
Prototypes
Qualification and Reliability Testing
Mil-Spec Products

Figure 1.3.3.4-1. Mil-Spec Testing Removed from the Critical Path of Prototype Development

1.4 Summary and Recommendations

Target System Recommendation—ATR has been identified as the preferred and most challenging target system for RASSP implementation; ATR ranked first, comm second, and EW third in our evaluation.

An ATR can clearly be partitioned into an analog front end, digital front end and an embedded processor with mass memory. A goal will be to move the embedded processing as far forward as possible. Generally, the embedded processor performs numeric and symbolic processing in an ATR system.

We recommend that the RASSP program accept the current ATR capability and establish it as the baseline for RASSP implementation. The performance improvements and increases in ATR functionality can be treated as model year upgrades. The baseline model can be an ATC. Each ATR upgrade will increase the performance, the number of users, or the functionality of the ATR. The baseline ATC/ATR is for air support purposes. The three upgrade options are, more platforms, precision strike, and reconnaissance/surveillance.

ATR is of interest to many users. A clear path and many applications for an ATR system are projected. We anticipate that ATR technology will mature and flourish by the end of this decade. It is therefore an excellent candidate for RASSP implementation.

System Design—This study emphasizes the need for a very structured concurrent product development process. This process and an associated procedural formalism based on F^3 practices are described and recommended. System-level design tools required to implement this process, particularly tools for requirements tracking, tradeoff analysis, system-level modeling/simulation, system hardware/software partitioning, system test generation, and documentation, are just emerging; and they need integration and usage experience to gain acceptance. They are critical to the requirements capture, performance capture, and high-level description tasks of the product development process.

A system-level simulation/emulation/hardware testbed is recommended as the basis for a system design environment. This testbed would be the cornerstone at the critical juncture in design where software and hardware partitioning are performed.

Hardware Design—Hardware design and hardware technologies, i.e., circuit and assembly, MCM, ASIC, are better defined and more mature than the system design area. Therefore, hardware technology is not viewed as an issue for RASSP (although it will be imperative to take advantage of new technologies such as GaAs ASIC, photonic interconnects, and new packaging being developed under ASEM programs under DARPA). The success of RASSP will be realized through the generic signal processor architecture and its openness and upgradability for new hardware and software; for example, the Touchstone architecture that we examined can be enhanced by high performance co-processors within the same hardware and software configuration.

Hardware design tools/methodology and their upkeep are the main hardware design concerns. The most critical needs are in the areas of detailed hardware design, PCB design/layout, and MCM design/layout. Besides needing better tool integration, tools capable of high frequency design analysis are needed. The tools for simulation of electromagnetic characteristics (i.e., characteristic impedance, reflections, cross-coupling) of the interconnects need to be improved and integrated with the more standard simulation and analysis tools. These tools should include tradeoff capabilities for optimization of speed, area, and power.

Several areas in the hardware design process need only be monitored for the progress of commercial developments. These include PLD/FPGA/gate array capabilities and ASIC test compilation development.

Software Design—The challenges for RASSP are particularly demanding in three software design areas: rapid prototyping/upgradability, affordability, and very high performance in real-time. New methods and new tools for software design are strongly recommended. A new methodology for software reuse is recommended and is a key to affordable RASSP (see Subsection 4.1). Software development technologies to be developed are identified (see Tables 2.4-1 and 2.4-2 of Subsection 2.4).

Software design issues critical to RASSP are, lack of architecture specification languages and domain specific tools for signal processing. The ongoing DARPA program Domain Specific Software Architecture (DSSA) addresses these two issues for different domains, namely, control and navigation. DSSA will provide a methodology for RASSP software design; however, the architecture specification language and signal processing domain specific tools need to be developed under RASSP.

Software areas only needing monitoring and/or new CASE tools and/or standards for specific domains are: architecture analysis/simulation, validation, verification and test, programming languages, embedded operating systems, and microcode development.

Information Infrastructure and Data Base—The recommended approach to the information infrastructure for RASSP is a federated network. The four basic data bases, engineering, manufacturing, management, and logistics are networked for easy interchange and sharing of cross-functional tools and of data processing services for reuse, traceability, configuration management, and change notification. Each area (or domain) could be supported by a different framework.

The most important recommendations pertain to data models and tracking development of the F-22 Integrated Weapon System Data Base. The data models for signal processing need to be standardized. The RASSP program should develop a signal processing application protocol that is consistent with CFI and PDES. CFI and PDES standardization efforts should be focused and accelerated in the direction of RASSP.

The areas that require tracking and/or research focus and stimulation are: data services, process services, cross-functional tools, reuse, traceability, configuration management, and change notification.

Testing and Evaluation—Testing technology has recently undergone a revolution due to the sky-rocketing complexity of systems, modules, and components. This increasing complexity manifests itself in the increased number of components, increased number and types of interfaces, varied technologies, mixed digital/analog designs, sophisticated functionality, etc. In contrast, the affordability drive of RASSP requires test simplicity/uniformity, high fault-tolerance, and a large proportion of self-test and performance monitoring.

The most important problems for RASSP testing are, test requirement and specifications, test program generation, and testability analysis. The testability analysis is limited on both system and chip levels to topology and structure information only. The desired approach is to include testability in the detailed tradeoff analysis at the system level for both hardware and software. The design for testability and accessibility should be established and followed. A recommended approach is a comprehensive testability analysis with multiple-domain information models at higher levels and signal processing testability guidelines at more detailed levels.

Test program generation is the second area requiring substantial RASSP participation. In spite of the ongoing programs (VTEST, TISS, ABET, and SS/REC) and standards (WAVES), the capabilities for tester independence and rapid development through reuse do not exist. The RASSP goal should be to generate the signal processing domain specific modeling approach and develop a model library leveraged by the ongoing programs.

The third area needing significant RASSP support is in test requirements and specifications. Test requirements cannot be extracted or traced back to requirements in an efficient and consistent way. The solution is to develop an integrated test system environment, use test requirement models to support analysis, and implement domain-specific requirement reuse.

Other testing recommendations include: supporting standards, encouraging vendor cooperation, and supporting research activities. The new standards are required for design and test beyond the IEEE 1149.1 into the analog, military, full-scan and other domains. Test equipment needs to be standardized especially in the area of MCM, HDI, and system testing. Standards and vendor cooperation are needed in areas such as test simulation (especially analog and mixed digital/analog), ASIC and board test equipment, software test /verification tools, and "known good die" determination for MCM and HDI packaging.

Finally, existing test synthesis tools are inflexible and rigid thus discouraging performance tradeoffs. RASSP should support research in this area (partially in progress) at universities, MCC, and industry.

Manufacturing, Design/Manufacturing Interface, and Cost—Our study results have stressed the view that manufacturing must be included as a co-participant in the overall concurrent product development environment for RASSP. We see advanced, automated manufacturing technology playing a key role in the RASSP implementation effort. It will be a risk reducer through fault avoidance in the assembly of fine-pitch advanced technology boards; and it will be a cost reducer through its quicker assembly time and implementation of best commercial cost-avoidance practices such as just-in-time stocking, use of group technologies/parts, and activity based costing. This manufacturing focus is mainly concerned with printed circuit board assembly because we have assumed that ASICs, ASEMs/MCMs, and even the fabrication of the multi-layer printed circuit boards themselves, will be viewed as vendor supplied parts, these parts acquired through a vendor-networked infrastructure.

The manufacturing organization required for RASSP is a new, vital organization characterized by: (1) high automation, planned for high volume assembly, yet amenable to lot sizes of one; (2) tight coupling to design so single prototype units can be introduced and built "on the fly"; (3) designed to accommodate change by employing agile, flexible manufacturing techniques; and (4) implemented in an organization committed to concurrent engineering/integrated product development "best practices" and to ongoing process improvement.

Manufacturing must be a full participant in the RASSP cross-functional process and must participate throughout. Producibility considerations must be part of this process from beginning to end.

The design-manufacturing interface must be a real-time, bi-directional, networked interface with CAD, CAM, and CAT tools linked and with common access to an integrated product development (IPD) data base. Development of this network and data base should be a part of the RASSP implementation phase. Recommendations for standards to implement this environment are: (1) Choose CAD, CAM, and CAT tools available on multiple platforms; (2) At a minimum, provide facilities for binary file transportability; (3) Encourage the use of non-proprietary file types such as DOS ASCII or postscript to facilitate file sharing between applications; and (4) use a combination of NFS, Ethernet, and TCP-IP for wide area networking.

We have recommended using "best commercial practices" as much as possible in the RASSP implementation phase. These practices include just-in-time parts procurement and stocking, on-line parts storage, certified inventory storage through qualified suppliers, eliminating incoming parts inspection, buying parts from qualified manufacturers, performing qualification and reliability testing mainly at the box or assembly level, using statistical process control and continuous process improvement methods, using activity based costing techniques, ensuring that field support is linked into and is part of this hole process, and establishing metrics so that cost and performance can be continuously monitored

A preference for using a commercial production facility for the RASSP implementation phase has been stated. If this cannot be accomplished, then we would recommend setting up a separate advanced assembly line for RASSP, apart from but in parallel with, a similar commercial line. The synergism between these two lines would still be beneficial to both and a risk reducer to the RASSP program. Our long-range hope is that the barriers between military and commercial businesses can be overcome or eliminated.

Detailed discussion of each of the above areas is provided in Section 2 and a further expansion of important findings and conclusions is given in Section 3. Section 4 examines recommendations for specific software and hardware developments and Section 5 comments on the implications for further research. A compendium of related work is included as Appendix A, and we have also attached a Honeywell proprietary addendum, "Model-Year Planning for Next-Generation Digital Avionics Signal Processors."

Section 2
Detailed Technical Results

2.1 Target System Selection

The objective of the system selection task was fourfold:

- To discuss the advantages and drawbacks of using the RASSP approach to develop a signal processing system
- To establish an approach for selecting a target system for RASSP
- To evaluate an ATR system, EW system, and Communication System based on the established approach
- To discuss which classes of system could benefit most from the Model Year approach

The following sections detailed our findings on these four objectives.

2.1.1 Advantages and Drawbacks of RASSP

The major advantages of the RASSP approach on the target systems result from the Model Year concept. System performance will steadily improve. The cost of upgrades should decrease. Insertion of state-of-the-art hardware, software and algorithms should keep the system on the technology edge. A standard system architecture across the military services is possible, thus potentially lowering cost. Dual use (i.e., military and commercial) of the target system's technology would broaden the application base and help reduce its cost. Industry's ability to upgrade systems in response to new challenges and threats should be more rapid.

The RASSP approach, however, is not without drawbacks. A RASSP design may waste precious resources, e.g., bus bandwidth, since growth allowance must be factored into the initial design. Military systems typically demand tight power consumption, volume, and weight requirements, which can be met only with a custom design. Initial relaxation of the system requirements may alleviate this problem. Another potential drawback is a higher initial development cost. This could be acceptable to companies if there is some guarantee of follow-on business for model year upgrades.

One concern associated with the RASSP approach relates to setting standards. Standards are no doubt important to the RASSP approach, but they must be set carefully and timely, and preferably be open standards. Periodic revision and upgrading of standards should be held regularly preventing technology obsolescence.

Lengthy military parts qualification, systems certification, and excessive documentation requirements represent potential delays for the RASSP approach. The RASSP program should explore ways to shorten these efforts to facilitate rapid turnaround.

2.1.2 Target System Selection Approach

The selection of a target system for this study was approached through a tradeoff analysis. Each candidate system was evaluated in six areas that relate to the success of a RASSP demonstration and ensure a sustained RASSP methodology even after the implementation phase. These six areas were: (1) U. S. Critical Need, (2) System, (3) Technology Maturity, (4) Manufacture, Production, and Field, (5) Model-Year Upgradability, and (6) Logistics.

The rationale in choosing these six areas was as follow. U. S. Critical Need determines the importance of the system among the DoD users. Broader DoD applicability also implies higher production volume. The system area relates to the target system's architecture and its compatibility with the proposed RASSP architecture of open, scalable interfaces, analog and digital front ends, and the embedded processor. A clear partition and proper interface of the three components will facilitate the system's model year upgrading. Technology maturity relates to the state of the art of the target system. The technology should be mature but possess growth capability. Manufacture, production and field essentially evaluates the affordability of the target system for production in a flexible factory. Model year upgradability depicts the options of a target system for improving its performance without major rework. Logistics addresses the seamless environment that provides communication and openness among different vendors, manufacturers.

A list of criteria was defined within each of the six areas for evaluating the target systems. Associated with each criterion is a rule for scoring that target system, the maximum possible score, and a weight. The weight, which normalizes or balances the relative importance of each category, was applied to the score in computing the figure of merit of a target system. The figure of merit is the sum of the weighed scores. Table 2.1.2-1 lists all the criteria, the maximum scores and the corresponding weights. A maximum of 208 points could be calculated for a target system.

The following describes each of the criteria:

Critical need and demand:

- Agencies—It counts the number of defense and space related Government agencies, such as Air Force, Army, Navy, NASA, ORD of CIA, that need the target system. Scoring method was 1 point per agency with a maximum of 5.

- Platforms—Another measure of the importance of the target system is the number of platforms on which it could be installed. Platforms include fighter aircraft, reconnaissance aircraft, bomber aircraft, helicopter, missile, satellite, tank, APC, other ground vehicles, ground stations, troop commanders, and warships. Scoring method was 1 point per platform up to a maximum of 10.

System/architecture:

- Component:—This criterion evaluated the target system for its ability to be efficiently and effectively partitioned into the digital, and analog front ends, and the embedded processor. Effective partition refers to a distinction among the three parts, and efficient partition refers to a clear cut among the three parts. The scoring was based on 2 points for each partition in the analog/digital front end, and digital/embedded processor; 1 point each for a clear cut in the analog/digital front end and digital/embedded processor. A maximum of 6 points could be scored.

- Inter-Operability—This criterion measured whether standard interfaces between the three components could be defined. A secondary evaluation was whether standard interfaces among the modules within each component could be well-defined. The scoring was 2 points each for the interfaces between the components and 1 point each for the interfaces within each component. A total of 7 points could be scored.

- Modularity—Modularity is concerned with the decomposition of each component into upgradable modules such as MCMs, LRUs, ASICs, chip sets, and software modules. The scoring method was 1 point each if the component could be modularized. Maximum score was 3.

- Dual Use Parts—This criterion measured whether the technologies applied in the target system could be used in both commercial and military applications. Four points were allocated if commercial parts were available to be used in the target system; 3 points for the eventual commercialization of the target system for industrial or commercial use. A total of 7 points could be scored.

Technology maturity:

The evaluation of the criteria in this category was based on a scale of one to five.

- Architecture—This criterion measures whether the architecture of the target system supports the processing requirements of that system.

- Hardware—Hardware technologies include memory, chip set, CPU, co-processor, packaging, buses, network, etc. The question is: does existing hardware technologies support the processing requirements of the target system?

- Software—Software technologies include an operating system (multiprocessor and real-time), database, language, software design methodology (CASE), and domain-specific software architecture. This criterion is an assessment of the maturity of the software technologies that are needed by the target system.

- Algorithm—Algorithm criterion evaluates the state of the art signal processing algorithm to determine whether it can meet the performance specified in the target system requirement.

Manufacture, production, and field:

The criteria in this category are as follows:

- Cost—This criterion evaluates whether or not the cost per unit of the target system is sensitive to the production volume If it is very sensitive, score 2 points; if somewhat sensitive, score 5 points; and if not sensitive, score 1 point.

- Existing product—This criterion evaluates whether or not the target system is an existing product. New standards and practices would be difficult to impose on existing manufacturability facilities. Therefore score 2 for non-existing products and 0 for existing ones.

- Testability—This criterion evaluates the testability of the target system. It determines the ease of, and requirement for testing the performance of the target system, and the individual components on a scale of 0 to 3.

Model year upgrade:

There are nine criteria under the model year upgrade. The first three are analog front end, digital front end, and the embedded processor, which addresses the possibility of upgrading the different components. They are scored on a relative scale of 0 to 5. An important criterion is the life cycle of the target system, which evaluates the life of the target system and the need for periodic upgrading. The score ranges from 0 to 10. The next two, which are improvement in accuracy and increase in functionality, address the upgrade possibility on different aspects of the target system. The last three criteria address the possibility of upgrade in form, fit, weight, and power requirements. Since they could be related to the first three criteria, the score is on a scale of 0 to 2.

Logistics:

- Teaming—This criteria evaluates the need for teaming with different vendors and manufacturers. It scores on a scale of 0 to 5 based on the relative number of vendors to be teamed with.

- Database—This criterion evaluates the existence of a database for the use in the target system. Score is from 0 to 5 based on the number of data base available.

- Standard—Similarly, this criteria evaluates the presence of any standard for the target system. It also determined whether or not the industry abides by the standards.

The six major areas of evaluation criteria were viewed as almost equally important for achieving a successful demonstration of the RASSP methodology. Therefore, the weighting of each of the criteria was applied such that the total possible score for each area was approximately equal. Note that the technology maturity area is weighted low. The reason is that technologies are improving at a great pace. An advancing technology, required for the target system and not far from insertion, could catch up fairly easily. The important element is whether the target system allows growth and model year upgrading. That is why the weight on the model year upgrade area is relatively high, a maximum of 46 points. The two areas, however, average close to the mean score of the 6 areas, which is 34.

Table 2.1.2-1. Target System Selection Criteria

		Max. Score	Weight
Critical Need & Demand (37.5)	Agencies	5	2.5
	Serviced Platforms	10	2.5
System (34.5)	Component	6	1.5
	Inter-operability	7	1.5
	Modularity	3	1.5
	Dual Use Parts	7	1.5
Technology Maturity (20.0)	Architecture	5	1.0
	Algorithm	5	1.0
	Software	5	1.0
	Hardware	5	1.0
Manufacture, Production, Field (32.5)	Cost	5	2.5
	Existing Products	2	2.5
	Testability	3	2.5
	Maintainability	3	2.5

		Max. Score	Weight
Model-Year Upgrade (46.0)	Analog	5	1.0
	Digital	5	1.0
	Processor	5	1.0
	Life cycle	10	1.0
	Accuracy	5	1.0
	Functionality	10	1.0
	Size / Form	2	1.0
	Weight	2	1.0
	Power	2	1.0
Logistics (37.5)	Teaming	5	1.5
	DataBase	10	1.5
	Standards	10	1.5

Apply Weight to normalize / balance the relative importance of each category.

Figure of Merit = SUM (Score$_i$ * Weight$_i$)

Max. weighed score = 208.0

2.1.3 Target System Evaluation

Using the target selection approach described above, three target systems, ATR, EW, and Communication were evaluated. The results of this evaluation are discussed in this section. First, the functions of the three systems are briefly described

An ATR system scans the area of interest, normally a battle field, and identifies for the operator all the real targets (with location, and classification) within the sensor's field of view. As an example, LANTIRN is considered an ATR system. EW systems detect all incoming threats aiming at the platform and provide the operator with threat warning signals. As an example, IRST is an EW system. Communication systems provide secure, non-interfered communications with remote locations.

The evaluation scores are listed in Table 2.1.3-1. The ATR system had the highest score (157), the communication system was second with 140 points, and the EW system was third with 135 points.

Table 2.1.3-1. Target System Selection

		Max. Score	Weight	ATR		E W		COMM	
				Score	Wt. Sc.	Score	Wt. Sc.	Score	Wt. Sc
U.S. Critical Need	Agencies	5	2.5	5	12.5	3	7.5	4	10.0
	Serviced Platforms	10	2.5	9	22.5	4	10.0	8	20.0
System	Component	6	1.5	6	9.0	4	6.0	2	3.0
	Inter-operability	7	1.5	7	10.5	5	7.5	3	4.5
	Modularity	3	1.5	3	4.5	2	3.0	1	1.5
	Dual use parts	7	1.5	7	10.5	4	6.0	4	6.0
Technology Maturity	Architecture	5	1.0	3	3	4	4.0	5	5.0
	Algorithm	5	1.0	2	2	4	4.0	4	4.0
	Software	5	1.0	2	2	5	5.0	4	4.0
	Hardware	5	1.0	2	2	4	4.0	5	5.0
Manufacture, Production, Field	Cost	5	2.5	4	10.0	2	5.0	3	7.5
	Existing Products	2	2.5	2	5.0	1	2.5	1	2.5
	Testability	3	2.5	2	5.0	3	7.5	3	7.5
	Maintainability	3'	2.5	3	7.5	3	7.5	3	7.5
Model-Year Upgrade	Analog	5	1.0	4	4.0	5	5.0	5	5.0
	Digital	5	1.0	5	5.0	4	4.0	3	3.0
	Processor	5	1.0	4	4.0	3	3.0	2	2.0
	Life cycle	10	1.0	10	10.0	8	8.0	7	7.0
	Accuracy	5	1.0	2	2.0	3	3.0	3	3.0
	Functionality	10	1.0	10	10.0	5	5.0	2	2.0
	Size / Form	2	1.0	1	1.0	2	2.0	2	2.0
	Weight	2	1.0	0	0.0	1	1.0	2	2.0
	Power	2	1.0	0	0.0	2	2.0	2	2.0
Logistics	Teaming	5	1.5	5	7.5	3	4.5	2	3.0
	DataBase	10	1.5	3	4.5	5	7.5	7	10.5
	Standards	10	1.5	3	4.5	7	10.5	7	10.5
Total					157.0		135.0		140.0

Figure of Merit = SUM (Score$_i$ * Weight$_i$)

Another analysis of the results was conducted to assess the strength and weakness of each system for RASSP demonstration. The weighed scores in each of the six areas were summed. A comparison of the scores in the same area indicated the differences among the three target systems.

The bar chart in Figure 2.1.3-1 shows the relative strengths and weaknesses of each system when compared against each other. Note that manufacture, production, and field did not stand out as a discriminator. This does not mean that this area is not important to RASSP methodology. Rather it indicates that similar difficulties and problems are likely to be faced by all three systems.

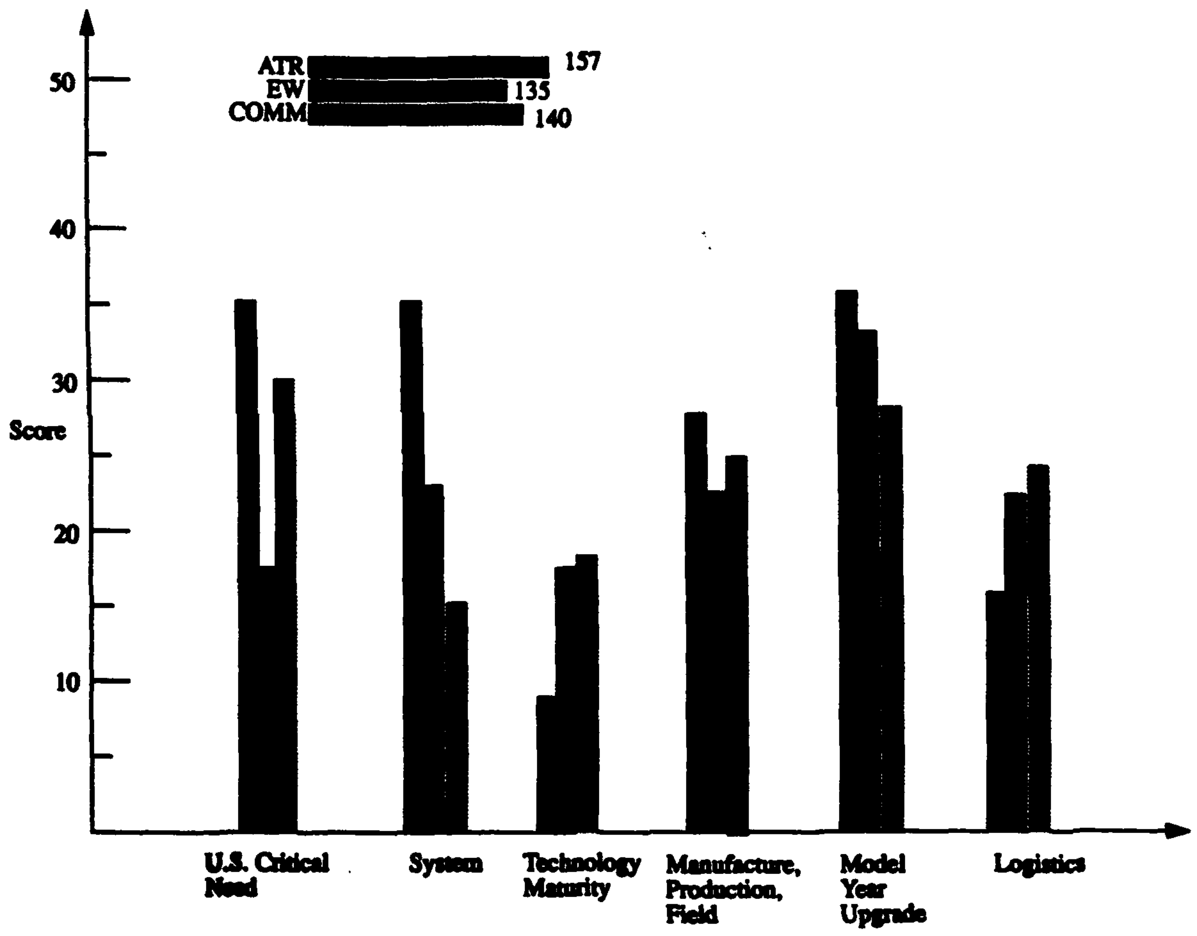


Figure 2.1.3-1. Relative Strengths and Weaknesses of RASSP Target Systems

The following discussion addresses the strengths and weaknesses of each system. The strengths of an ATR system for RASSP are in the areas of U.S. critical need, system, and model year upgradability. Every military service needs an ATR to enhance the success of a mission while minimizing losses. ATR systems can be installed on almost all the platforms mentioned in the criteria. This will yield a higher volume of production. An ATR can be partitioned into an analog front end interfacing with one or more sensors, a digital front end involving computational intensive sensor preprocessing, and the embedded processor for signal and data processing. It is very important to recognize that the partitioning and modularity of the sensor preprocessing digital front end and of the embedded high-performance computing (HPC) modules associated with the embedded processor afford tremendous flexibility when tailoring the system to specific and potentially multiple sensors. This is a real strength of the proposed generic RASSP architecture and one that can be fully exploited by the Touchstone architecture. It also follows from this flexibility that ATR technology can be applied to both military and commercial applications. Character recognition, industrial robotics and vision, earth resource analysis, home and building surveillance, automotive synthetic vision and collision avoidance, and a host of medical applications are some of the dual commercial applications. Lastly, numerous model year upgrades are possible for an ATR system, and this would be a key benefit to RASSP.

ATR, on the other hand, is weak in the technology maturity and logistics areas. Advanced algorithms, which have demonstrated their feasibility on limited cases, exist. Their real time implementation into an ATR system is limited by the availability of adequate processing. In other words, the computational requirement tends to always exceed the processing power of the hardware. As the system architecture and processing advance, the gap between the supply of high-performance processors and demand for high-throughput processing will become narrower.

Another ATR weakness is in the area of logistics. In particular, ATR database information and standards are lacking. The main reason is that no ATRs are in production yet. Regarding standards, much research and development is still going on, and there is little consensus on standards in this area among the Government agencies and industry. Database information, especially on the second generation FLIR sensor, is very rare and inaccessible to ATR algorithm developers who need it for algorithm performance evaluation and testing.

The EW system's scores in most of the six areas are in between those of other two systems. EW has a strength in the model year upgrade area. Recent development in signal processors has allowed more computational intensive algorithms to be put into EW systems. This is a major benefit for its model year upgrading. However, EW suffers weakness in its U.S. Critical need as compared with the other candidate systems. The Air Force and Navy are the two main users, and the commercial application of EW technology is not high.

Communication systems are strong as a U.S. critical need. Communication systems are needed by every military force and many other agencies. Commercial use of communication technology is also tremendous. Also, communication system technology is relatively mature. Models and communication networks are well-established within the defense and commercial worlds. Security is probably the last system hurdle to overcome. The major communication system weakness for RASSP is in it architecture. The distinction between the analog front end and the digital front end is not very clear. Advances in digital signal processing has crossed into the analog front end component. Thus the interfaces between the two components is manufacturer dependent. A similar situation occurs between the digital front end and the embedded processor.

In summary, the ATR system is viewed as the most challenging system for the RASSP implementation phase. It possesses the most strengths but also the most weaknesses. Assuming the ATR's weaknesses can be overcome or its risks tolerated, it should be the strongest candidate for RASSP implementation.

2.2 System Design

The application specific electronic system (ASES) design for RASSP should be viewed as part of a broader concurrent product development process. This broader, concurrent process can establish the environment for rapid and seamless development of a product, including both hardware and software, from concept to fieldable prototype and/or fully qualified product and to support that product's model year updates. This assertion infers the concept of a very structured, highly integrated process for product definition and development together with the support of a integrally networked infrastructure.

In its entirety such a process does not exist today although we have a good sense of what it would be. However, elements of this process do exist. For example, ASIC design methodology, ASEM/MCM design methodology, and other elements are being developed. The overall "big picture" process is continuously developing and evolving. To tie all these elements together, a system design formalism is necessary. This formalism, described in the following section, will provide the consistency of product and process descriptions necessary to facilitate the overall concurrent product development process. More importantly, this formalism greatly facilitates a product's model year upgrading and the likelihood of achieving product and process interoperability. We will have more to say about this in the body of this report.

Why are we being so "tutorial" about this system development process and how does this discussion relate to RASSP implementation risks. We will offer three answers: (1) At that point in the hierarchical system design process where software and hardware implementation tradeoffs and system SW/HW partitioning are performed, there does not seem to be a well-defined common set of metrics that anchor this decision point; and subsequently these two processes (of software and hardware development) seem to be performed orthogonally, not linked in a manner that would help reduce design risk and improve design tracking, (2) The second reason is that the higher levels of concurrent product development methodology are still fairly immature and new tools are just emerging. Thus it is helpful if there is a broader context attempting to tie all these new tools and procedures together, aiming at the seamless product development environment envisioned for RASSP implementation. (3) Most importantly, this formalism is itself a risk reducer, not just for the RASSP implementation phase, but for product development and product upgrading in general.

2.2.1 Design Process Formalism

The design process formalism described here is not new but is fundamental to the engineering problem solving approach. The three basic steps that make up this process, definition, implementation, and verification, are repeated over and over again throughout the design process as the process progresses and the design problem is hierarchically decomposed.

- Definition—Given a set of requirements and associated data, define the design/problem and its parameters.

- Implementation—Develop a candidate design model or problem solution, and optimize it by iteration as required. This implementation step by its nature introduces the concept of hierarchy, the idea of an element composed of sub-elements.

- Verification—Verification has two important aspects: one is verification of the consistency of the design/problem representations as the design is hierarchically decomposed and/or integrated throughout the design process. The other aspect of verification is checking that an actual result from a step in the design process realizes the intended result. In both cases one iterates the process until a design/solution is completed and it is consistent throughout.

This design process formalism is diagrammed in Figure 2.2.1-1; namely, the concept of an element with inputs and outputs which can be decomposed into similar "child" elements also with inputs and outputs.

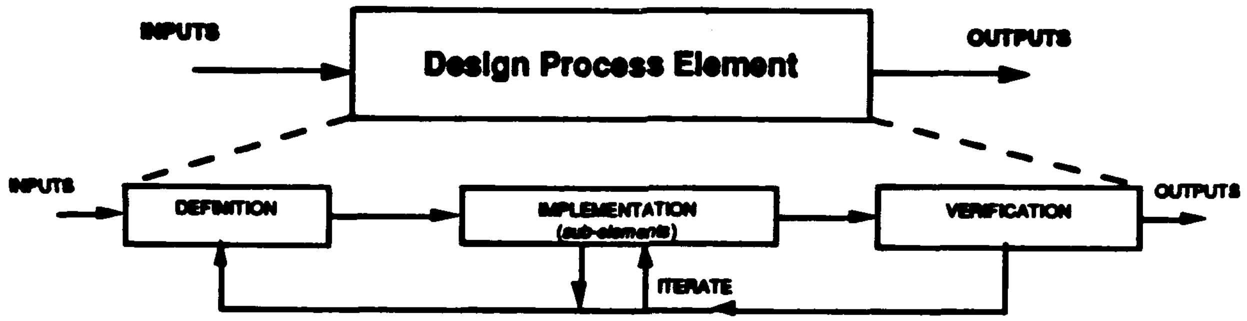


Figure 2.2.1-1. A Design Process Element and Its Decomposition

The design itself, both software or hardware, must be described in a canonical form that can provide a measure of the design's completeness. The design attributes of form, fit, and function (F^3) will be used as the specifications necessary to completely define a design. Again this is nothing new. The airline industry has been using F^3 practices for years, and the DoD has occasionally applied it to new systems.

- Form: The design's physical description; i.e., its makeup and composition, size, weight, power, material, packaging, lines of code, bits per word, etc.

- Fit: The design's interfaces to the outside world; i.e., its architecture or structural interfaces, buses, I/Os, connectors, standard interface formats.

- Function: The description of the design's behavior; i.e., what it does, timing, test, operating modes, software functionality, etc.

At any and all levels of a hierarchical design, each element must have form, fit, and functional descriptions for that element, whether hardware or software, to be completely defined.

When viewed in the larger context of concurrent product development, the attributes of form, fit, and function provide a concise basis for introducing the "ilities" into the design process right up front in the requirements capture and performance capture phases of the process. As the design is subsequently defined and hierarchically decomposed, consideration of the "ilities" on the design can be imposed and consistently traced throughout the design and configuration management processes. Design decisions thus include consideration of manufacturability, quality, testability, reliability, affordability, etc, as an integral part of the design process. And again this can apply to both hardware and software.

2.2.2 Concurrent ASES Design/Development Process

Figure 2.2.2-2 shows the top-level concurrent process flow for RASSP exploded out of an overview diagram of the ASES Design/Development Process to emphasize the equivalence of their parallel flows and major steps. For simplicity, no feedback or feed-forward paths are shown. Below each of the design steps in the lower diagram are listed the key tasks involved in that design step. There is a formal review associated with each of these steps.

Board, module, and software design flows are indicated as paralleling the ASIC design flow but branching off hierarchically above it. Back-plane/interconnect design is integrally related to the board and module design but is shown separately to emphasize its importance for high-performance RASSP processors. Data base and library development, including various levels of modeling, are critically important bottom-up functions impacting the overall ASES design/ development process. The board/ASES, module/ASEM, and ASIC design flows and model library development are outlined in Section 2.3, Hardware Design. Software design is covered in Section 2.4. The remainder of this section provides more detailed discussion of the initial system design tasks to illustrate the rigorous application of the formal F^3 design decomposition procedure.

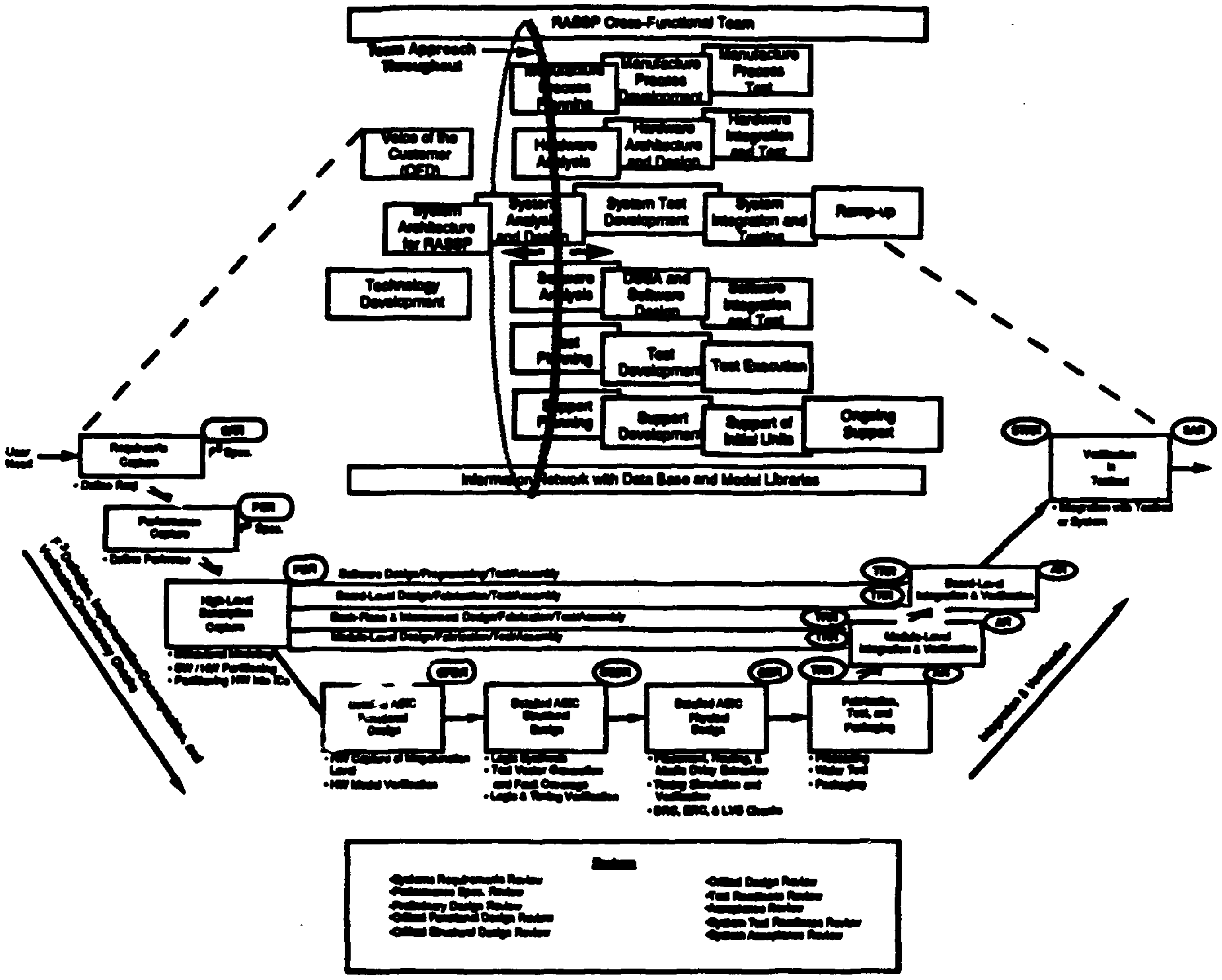
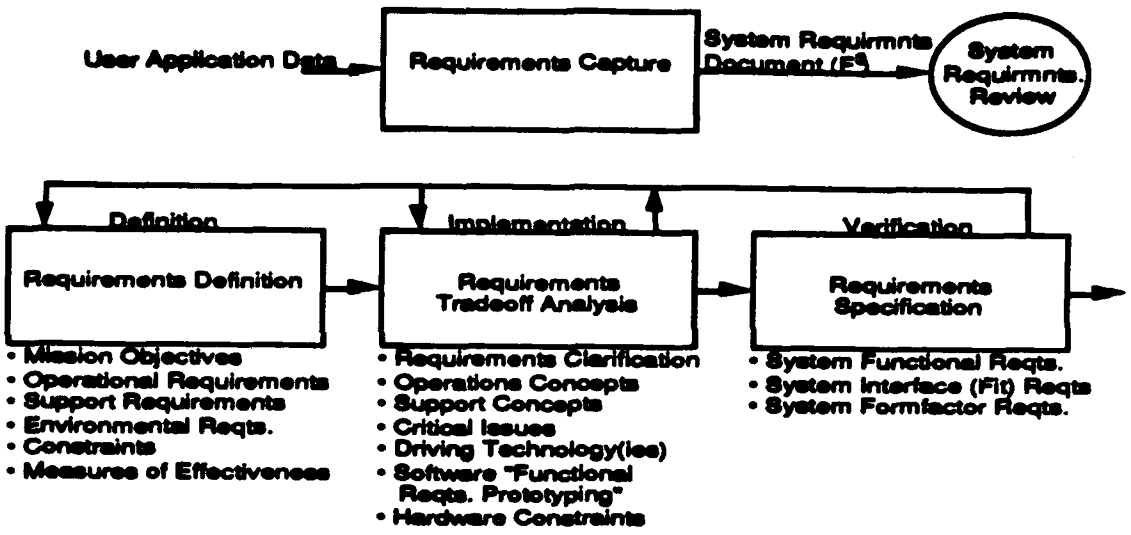

RASSP Cross-Functional Team
Team Approach Throughout
Manufacture Process Development
Manufacture Process Test
Hardware Analysis
Hardware Architecture and Design
Hardware Integration and Test
Voice of the Customer (QFD)
System Architecture for RASSP
System Analysis and Design
System Test Development
System Integration and Testing
Ramp-up
Technology Development
Software Analysis
DSSA and Software Design
Software Integration and Test
Test Planning
Test Development
Test Execution
Support Planning
Support Development
Support of Initial Units
Ongoing Support
Information Network with Data Base and Model Libraries
User Need
Requirements Capture
Performance Capture
High-Level Simulation Capture
Verification in Testbed
Board-Level Integration & Verification
Module-Level Integration & Verification
Detailed ASIC Structural Design
Detailed ASIC Physical Design
Fabrication, Test, and Packaging
Integration & Verification
Reviews

Figure 2.2.2-2. Concurrent ASES Design/Development Process with Detailed View of ASIC-Level Design/Fab/Test/Assembly Flow

2.2.3 System Requirements Capture

Interact with the customer and iterate with the customer's requirements to reach a mutually acceptable specification of system requirements. Figure 2.2.3-1 summarizes this task.


User Application Data
Requirements Capture
System Requirmnts Document (FS
System Requirmnts. Review
Definition
Implementation
Verification
Requirements Definition
Requirements Tradeoff Analysis
Requirements Specification
• Mission Objectives
• Operational Requirements
• Support Requirements
• Environmental Reqts.
• Constraints
• Measures of Effectiveness
• Requirements Clarification
• Operations Concepts
• Support Concepts
• Critical Issues
• Driving Technology(ies)
• Software "Functional Reqts. Prototyping"
• Hardware Constraints
• System Functional Reqts.
• System Interface (Fit) Reqts
• System Formfactor Reqts.

Mentor's System Design Station
Ascent Logic's Requirements Driven Design (RDD)
Honeywell's Avionics Requirements to Test Tracking System (ARTTS)
Honeywell's Architecture Tradeoff Tool (ATOT)
Spreadsheet

Figure 2.2.3-1. System Requirements Capture CAD Tools

2.2.3.1 Inputs—The user's requirements may be provided in many different ways and degrees of specificity. Near production this could be a "build-to" specification without much room for negotiation of requirements. However, in R&D type programs, the user is often less specific and only states guidelines or a combination of musts and wants leaving it for the system developer to formulate and propose his best definition of requirements.

For the RASSP program, DARPA suggested several candidate applications, the model year concept and the generic signal processor architecture (see Figure 1.2-1). Honeywell has proposed that this basic architecture can be utilized not only for front-end sensor processing but also for back-end display processing thus broadening its applicability and potential product volume. This concept is shown in Figure 2.2.3.1-1.

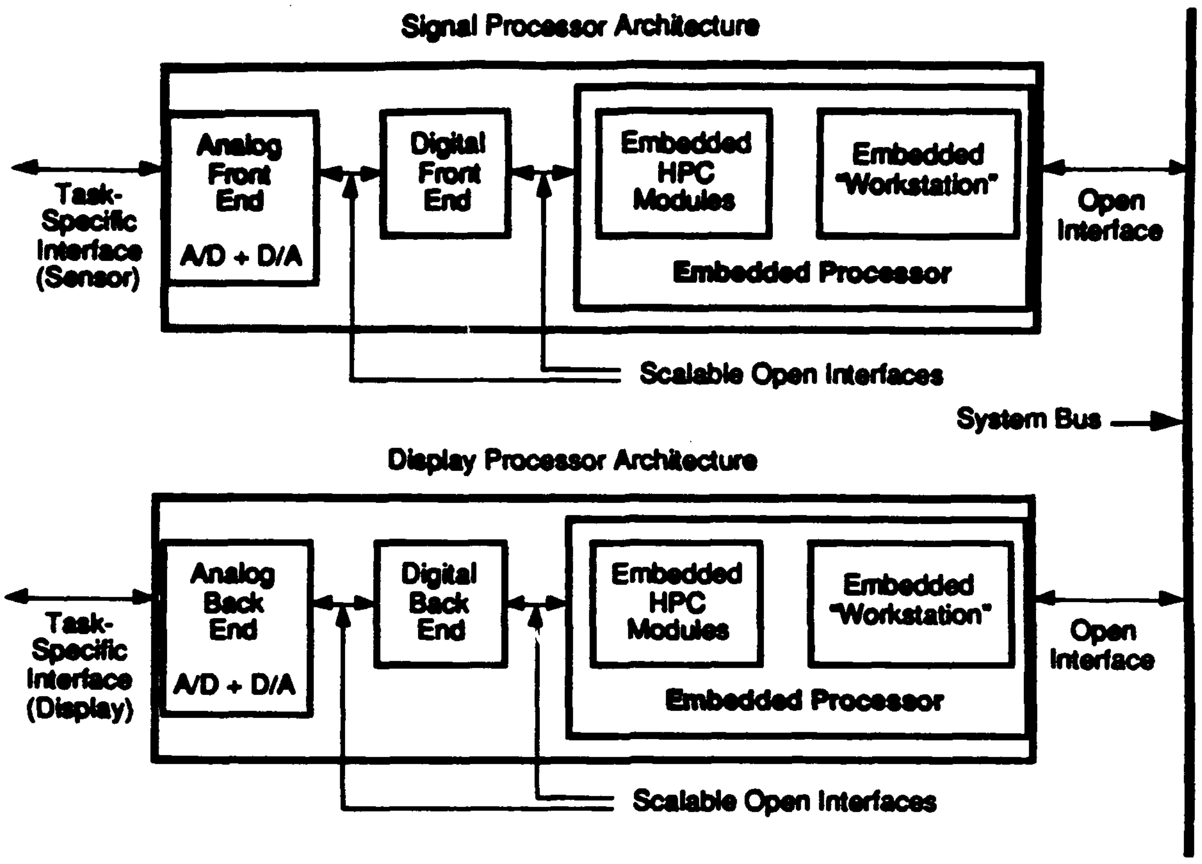


Figure 2.2.3.1-1. Honeywell Proposed Extended RASSP Architecture

2.2.3.2 Requirements Definition—From the very beginning, the formalism of the design process can be facilitated by defining the requirements within the context of form, fit, and function. Mission objectives, operational requirements, support requirements, environmental requirements, size constraints, etc., all can be sorted into F^3 requirements. For example, operational requirements for three major near term embedded processing system applications are summarized in Table 2.2.3.2-1.

Strategic requirements for the militarization of the Intel-DARPA Touchstone processor are summarized in Table 2.2.3.2-2.

Table 2.2.3.2-1 Processing Requirements for Three Major DoD Systems

System Attributes / Militarized Touchstone Opportunity	Signal Processing	Data Processing	LAN Stations/ Operators	System Features
JSTARS	200 GMops	11.1 GFlops	≤ 18	• Fault Tolerant • Secure • Scalable
AEGIS	~800 GMops	200 GFlops	~ 20	• Very Secure • Fault Tolerant • Scalable
Advanced AWACS	20 GMops	10 GFlops	≤ 18	• Fault Tolerant • Secure • Scalable

Table 2.2.3.2-2. Requirements for the Militarization of the Intel-DARPA Touchstone Processor

1) No new chip design—use the <u>exact</u> Intel design for core system

2) The system must be evolvable—upgrade cycle ≤ two years

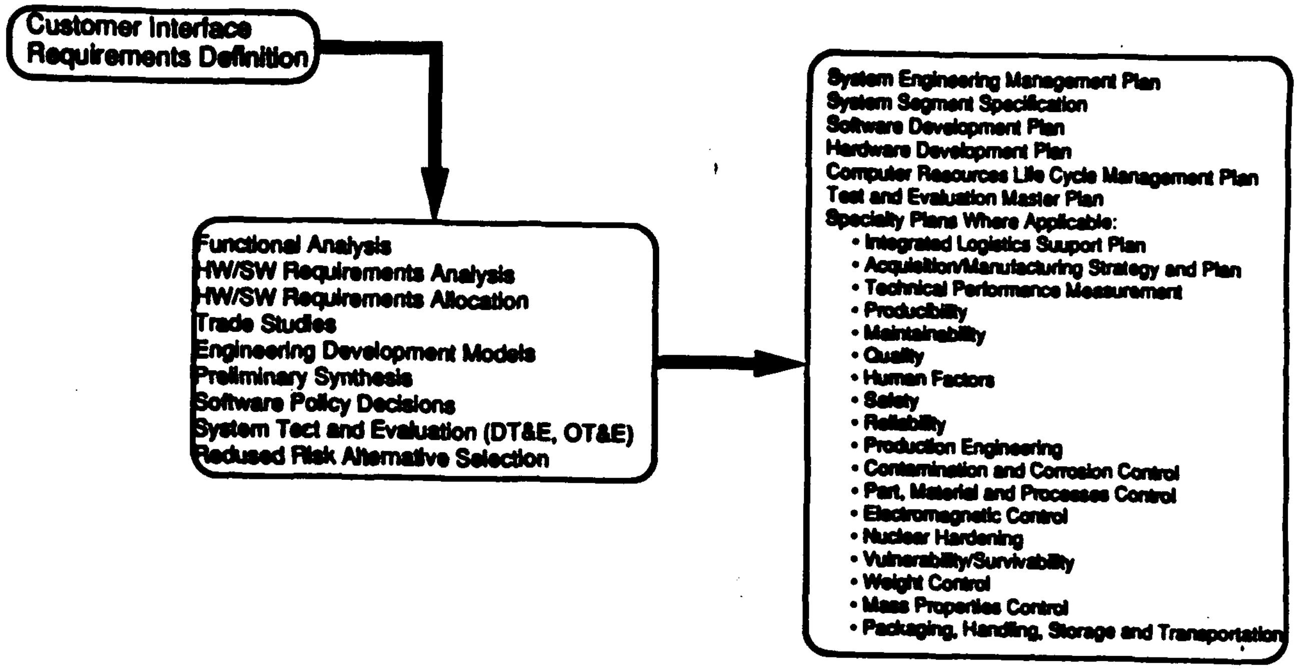
   - The packaging-cooling system will support higher thermal densities
   - The militarized systems will lag commercial introduction by ≤ six months

3) Military form factors will be followed:

   - SEM-E
   - Rugged cabinets

4) Future systems will require fault tolerance, security, and parallel utilities

2.2.3.3 Requirements Tradeoff Analysis—Requirements tradeoffs, as indicated in Figure 2.2.3.3-1, are the first in a hierarchical progression of analyses that further and further penetrate the design/decision process. The logical thread that links this design/decision process, as the process delves deeper into the system hierarchy, establishes the consistency that must pervade the entire process. If the tradeoff analyses and their decomposition can be performed in such a way so as to maintain the three intertwined threads of form, fit, and function, the process formalism and rigor of decisions can be clearly traced and documented. Table 2.2.3.3-1 lists some of these requirements tradeoffs and their related F^3 metrics.



Customer Interface Requirements Definition
Functional Analysis
HW/SW Requirements Analysis
HW/SW Requirements Allocation
Trade Studies
Engineering Development Models
Preliminary Synthesis
Software Policy Decisions
System Tect and Evaluation (DT&E, OT&E)
Redused Risk Alternative Selection
System Engineering Management Plan
System Segment Specification
Software Development Plan
Hardware Development Plan
Computer Resources Life Cycle Management Plan
Test and Evaluation Master Plan
Specialty Plans Where Applicable:
• Integrated Logistics Suuport Plan
• Acquisition/Manufacturing Strategy and Plan
• Technical Performance Measurement
• Producibility
• Maintainability
• Quality
• Human Factors
• Safety
• Reliability
• Production Engineering
• Contamination and Corrosion Control
• Part, Material and Processes Control
• Electromagnetic Control
• Nuclear Hardening
• Vulnerability/Survivability
• Weight Control
• Mass Properties Control
• Packaging, Handling, Storage and Transportation

Figure 2.2.3.3-1. The Analysis of Customer Requirements is a First Step in the Design Process

Table 2.2.3.3-1 Top-Down Requirements Drive the Product Development Process

Trade-Off	Related Metrics	Description
MCM vs. PWB	size, cost, speed, weight, power, thermal, reliability, schedule	Determine component interconect routing method.
S/W vs. H/W	size, cost, speed, weight, power, schedule, software metrics	HW/SW trade-off using VHDL and hardware modelers.
MCM Substrate	size, cost, speed, reliability, thermal	Substrate selection has a direct effect of speed, size and cost.
Chip Attachment	size, speed, cost, reliability	Typically only a few choices available, driven heavily by chip vendor market demand.
Test Techniques	schedule, time-to-test, cost, size, power	Testability usually specified only as coverage. Test equipment and time-to-test throughout life cycle must be evaluated.
ASIC Technologies	speed, power, size, cost, thermal	CMOS, ECL, GaAs etc. trade-off.
ASIC Implementation	size, speed, power, schedule, availability	Closely related to ASIC Technology. (gate array, standard cell, PLD, FPGA)
Packaging Approach	size, cost, speed, thermal	Coupled with MCM substrate and PWB type, will be driven by partitioning (signal I/O count) and thermal disipation requirements.
Thermal Dissipation	cost, weight, power, size	Most often cooling capability and thermal interface given by next higher level in the system.

2.2.3.4 Requirements Specification—Collating the final set of requirements into a specification document establishes the requirements spec for the subsequent design. Each of the selected/

established requirements should be compared against the original user requested needs to verify and confirm the extent to which these needs are met.

2.2.3.5 Outputs—The output of this first phase of the design process is a requirement specification agreed to by the user and the system developer.

2.2.3.6 Systems Requirements Review (SRR)—A Systems Requirements Review is held to formally sign-off on the agreed to requirements specification document.

2.2.3.7 CAE Tools for Requirements Capture—The development of commercial CAE tools to support the early, higher-level steps in the design/product development process is a fairly new thrust within the EDA industry. It has evolved from the advent of frameworks and the desire to provide CAD tool environments that can totally manage the design. At this early stage of tool emergence brochures describing new CAD tool products can be misleading. One is always wise to perform hands-on evaluations.

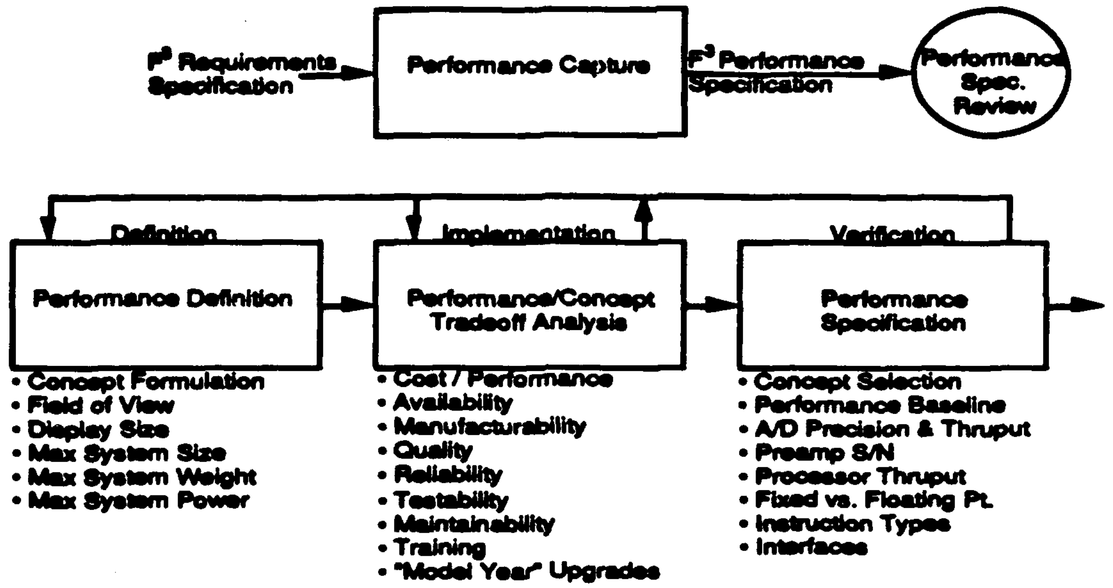
The basic capability needed to perform requirement tradeoffs is Kepner-Tregoe (K-T) type analysis; i.e., totalling weighted values for decision elements. This type of analysis works well using a spreadsheet format and it has proven useful for the selection of RASSP target system (see Section 2.1). Honeywell has an internally developed program, Architecture Tradeoff Tool (ATOT), that performs this type of analysis, but we prefer to report on commercially available tools.

For performing requirement capture and requirement tradeoff analysis, our CAD tool survey did not identify any vendor supplied tools that specifically offered K-T analysis or other unique functionality. Mentor's product description for its System Design Station comes closest stating, "Unified, automated environment for requirement capture, top-down system-level analysis, state machine animation, rapid prototyping, and simulation." The requirement capture may involve basic text editing, but Mentor claims to have added a tagging feature that could start implementing the formal design process that we are describing.

2.2.3.8 Issues/Risks—CAD tools for requirement capture must be evaluated in the context of the overall product development process. CAE industry activities in this area suggest that more tools will become available. It is likely that these tools will be developed to operate within a larger design management system. The main concern should be for continuity of data flow and compatibility for data exchange. CFI standards should help solve this problem. Availability of this CAD tool capability represents minimal risk to the RASSP implementation phase.

2.2.4 System/Subsystem Performance Capture

Transform the requirements specification into a performance specification. Interact with the customer and iterate with the requirement specification to reach a mutually acceptable system performance specification.



CAD Tools
Mentor's System Design Station
Ascent Logic's Requirements Driven Design (RDD)
Honeywell's Avionics Requirements to Test Tracking System (ARTTS)
Honeywell's Architecture Tradeoff Tool (ATOT)
Spreadsheet
SES Workbench

Figure 2.2.4-1 System/Subsystem Performance Capture

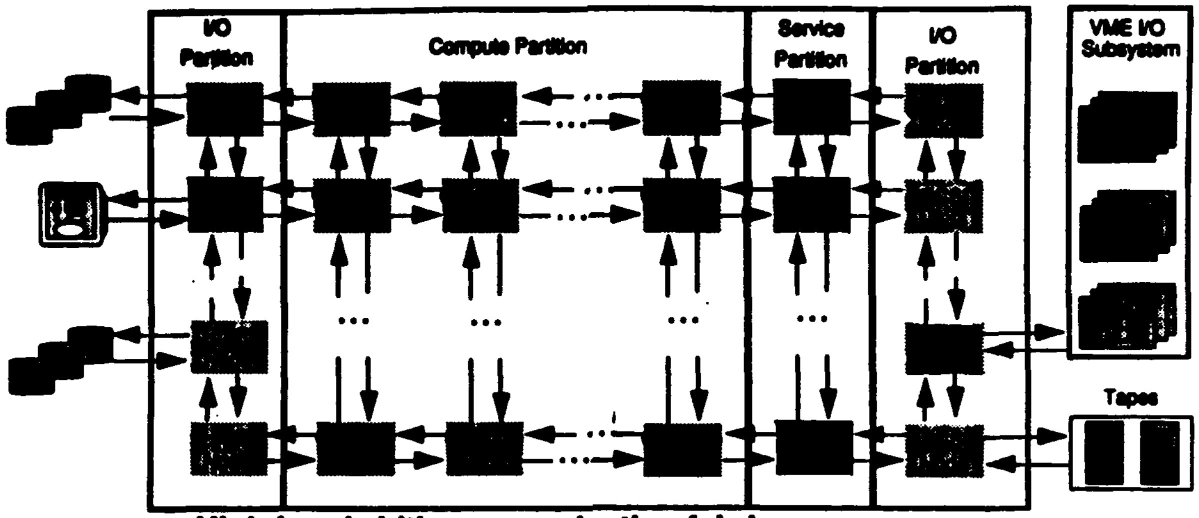
2.2.4.1 Inputs—The requirements specification, preferable organized in terms of functional requirements, form-factor requirements, and interface/fit requirements.

2.2.4.2 Performance Definition—Continuing from the Requirements Specification, the formalism of the design process can be maintained by defining performance within the context of form, fit, and function; i.e., a conceptual design's performance can be sorted into an F^3 performance specification.

As an example for this RASSP study, the generic signal processor architecture of shown previously in Figure 2.2.3.1-1 was examined in the context of a scalable two-dimensional mesh of militarized Touchstone processors. This 2-D mesh is shown in Figure 2.2.4.2-1. The scalable heterogeneous multiprocessor architecture can provide high-bandwidth performance with functionally optimized nodes, and it is open to continuous technology insertion.

Packaging form-factor projections for the advanced militarized Touchstone processor are shown in Figure 2.2.4.2-2. These advanced packaging features translate to lower DoD system size, weight, and power. Packaging characteristics must a part of performance capture.

Scalable heterogeneous multicomputer



- **High-bandwidth communication fabric**
- **Functionally optimized nodes**
- **Open to continuous technology insertion**

Figure 2.2.4.2-1. Two-Dimensional Mesh of Militarized Touchstone Processors

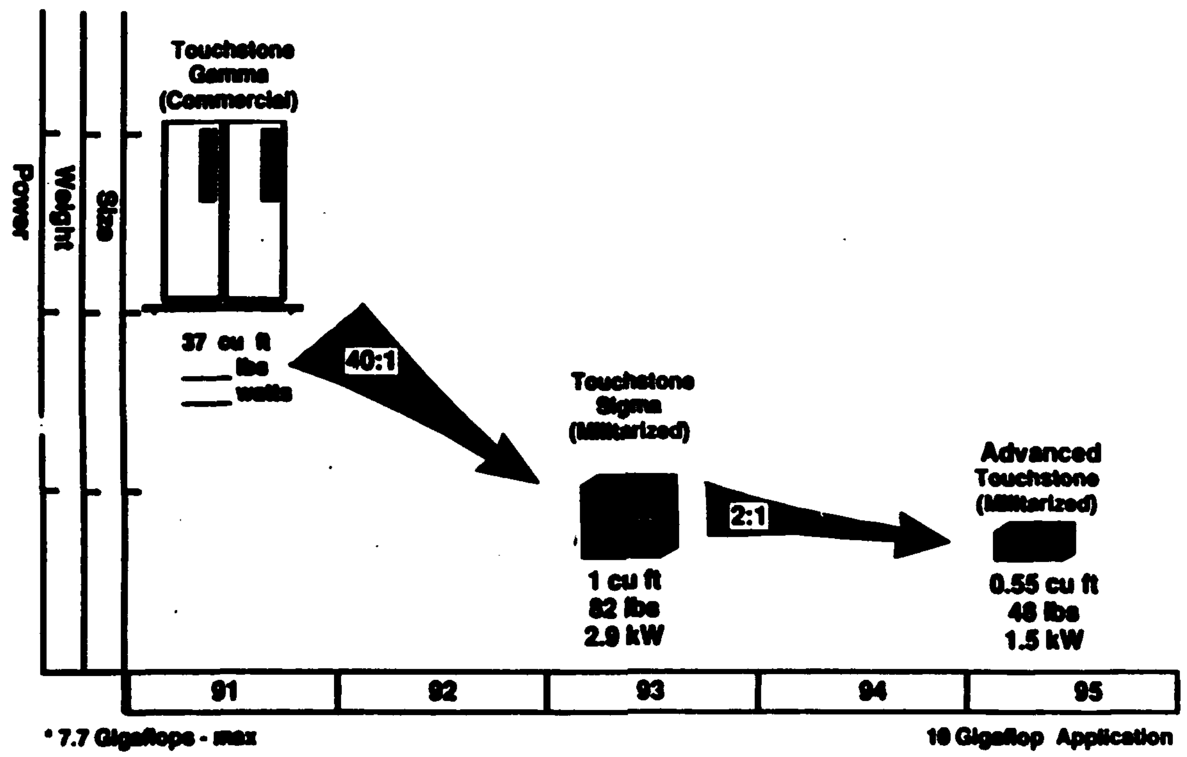


Figure 2.2.4.2-2 Advanced-Packaged Militarized Touchstone Form-factor Projections

2.2.4.3 Performance Tradeoff Analysis—Performance tradeoffs are the second in the hierarchical progression of analyses that further and further decompose the design/decision process. The logical thread that links this design/decision process as the process delves deeper into the system hierarchy establishes the consistency that must pervade the entire process. If tradeoff analyses and their decomposition can be performed in a custom that maintains the three intertwined threads of form, fit, and function, the process formalism and rigor of decisions can be clearly traced and documented.

2.2.4.4 Performance Specification—Collating the final set of performance goals into a document establishes the performance specification for the subsequent design. Each of the selected/established performance goals should be compared against the original requirements specifications to verify and confirm the extent that the requirements are met.

The model year concept introduces additional considerations for the performance specification. One can envision defining planned upgrades into the performance specification. Such upgrades for the Touchstone processor are illustrated in Figure 2.2.4.4-1. A procurement contract's terms and conditions would have to define the extent that the offerer is contractually obligated to fulfill these planned system upgrades. Traditional military contracting might provide for system upgrades as procurement options; but in the commercial world it is market competition that drives a company to upgrade their product.

Issue—How will government contracting practices be adapted to implement the model year upgrade concept?

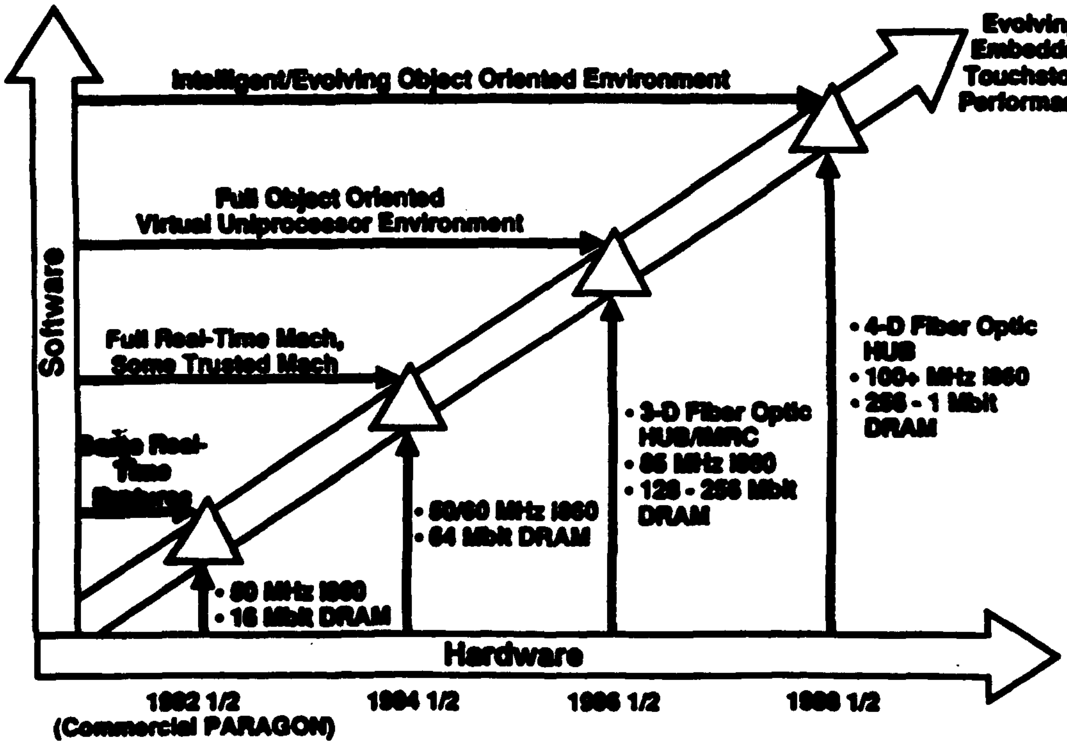

Evolving Embedded Touchstone Performance
Intelligent/Evolving Object Oriented Environment
Full Object Oriented Virtual Uniprocessor Environment
Full Real-Time Mach, Some Trusted Mach
Basic Real-Time
Software
• 4-D Fiber Optic HUB
• 100+ MHz i860
• 256 - 1 Mbit DRAM
• 3-D Fiber Optic HUB/MRC
• 85 MHz i860
• 128 - 256 Mbit DRAM
• 50/60 MHz i860
• 64 Mbit DRAM
• 50 MHz i860
• 16 Mbit DRAM
Hardware
1992 1/2 (Commercial PARAGON)
1994 1/2
1996 1/2
1998 1/2

Figure 2.2.4.4-1. Potential Model Year Upgrades of the Touchstone Processor

2.2.4.5 Outputs—The output of this second phase of the design process is a performance specification agreed to by the user and the system developer.

2.2.4.6 Performance Spec Review (PSR)—A System Performance Specification Review is held to formally sign-off on the agreed-to performance specification document.

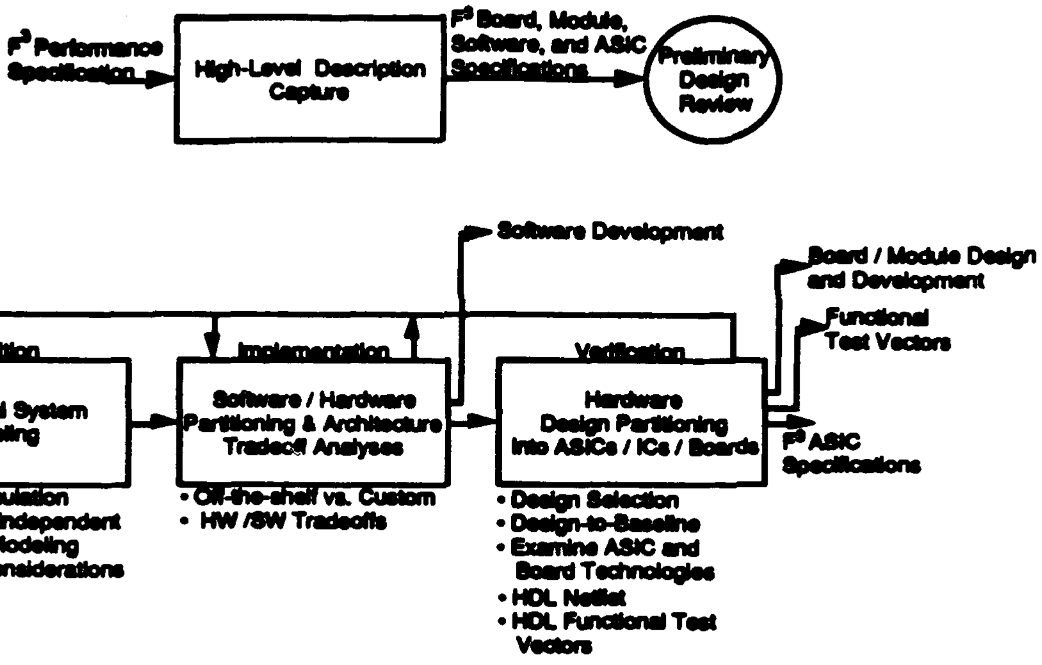
2.2.4.7 CAE Tools for Performance Capture—The CAE tools for performing performance capture are similar to those cited for requirement capture. The basic capability needed to perform performance tradeoffs remains Kepner-Tregoe type analysis; i.e., totalling weighted values for decision elements. This analysis could still involve a spreadsheet format or a program such as Honeywell's Architecture Tradeoff Tool (ATOT). Mentor's System Design Station would also still apply here.

A system modeling and analysis tool such as Scientific and Engineering Software's (SES) Workbench might be introduced into the system development process at this point to capture the seeds of function block performance. But it is at the following level of the design hierarchy that tools such as SES/Workbench and Mentor's System Design Station become particularly useful.

2.2.4.8 Issues/Risks—CAD tools for performance capture must be evaluated in the context of the overall product development process. CAE industry activities in this area suggest that more tools will become available. It is likely that these tools will be developed to operate within a larger design management system. The main concern should be for continuity of data flow and compatibility for data exchange. CAD Framework Initiative (CFI) standards should help solve this problem. Availability of this CAD tool capability represents minimal risk to the RASSP implementation phase.

2.2.5 High-Level Description Capture

The purpose of this task is to model the system at a high enough level to support concept tradeoff analyses leading to the partitioning of the system into a baseline conceptual design, as shown in Figure 2.2.5-1.


F3 Performance Specification
High-Level Description Capture
F3 Board, Module, Software, and ASIC Specifications
Preliminary Design Review
Software Development
Board / Module Design and Development
Functional Test Vectors
Definition
Implementation
Verification
Behavioral System Modeling
Software / Hardware Partitioning & Architecture Tradeoff Analyses
Hardware Design Partitioning into ASICs / ICs / Boards
F3 ASIC Specifications
• Design Formulation
• Technology Independent Functional Modeling
• Algorithm Considerations
• Off-the-shelf vs. Custom
• HW /SW Tradeoffs
• Design Selection
• Design-to-Baseline
• Examine ASIC and Board Technologies
• HDL Netlist
• HDL Functional Test Vectors

CAD Tools
Mentor's System Design Station
SES Workbench
i-Logix's Statemate
ADAS
Enhanced VHDL
Honeywell's Architecture Tradeoff Tool (ATOT)
Spreadsheet

Figure 2.2.5-1. High-Level Description Capture

2.2.5.1 Inputs—F^3 Performance Specification

2.2.5.2 Behavioral System Modeling—Behavioral system modeling starts the process of capturing a representation or abstraction of the design for subsequent analysis. The specifics of this step depend upon the tools used; but the step is the traditional "point of entry" to CAD and CASE. The task tends to be driven initially by functional performance (i.e., behavioral modeling) but the form and fit specifications must also be carried along to achieve the ultimate goal of software, hardware, and architecture definition. A combination of spreadsheet and simulation tools are required to accomplish these tasks. The initial goal is to formulate a conceptual design.

2.2.5.3 Software/Hardware Partitioning and Architecture Tradeoff Analyses—The specifics of this task are determined by our ability to identify specific steps and tools and detail this analysis process. For example, the steps could be identified as follows. Notice that the steps are expedited by prior methodology and existing data if we are in the model year upgrade phase of a product's life cycle.

1. Start with currently popular generic modular architecture

2. Evalur ˋroughput capability

3. Identify technologies capable of meeting throughput requirements

4. Explore off-the-shelf vs. custom and make vs. buy decisions

5. Analyze special adaptations to the architecture or software that may add performance value, cost, risk, etc.

6. Perform cost/benefit analyses as an extension of the tradeoffs analyzed during the prior performance capture design phase.

7. Establish the "design-to-baseline" for software development

Many tradeoff considerations need to be made at this point, a few of which were summarized in Table 2.2.3.3-1, previously. The ability to capture a hardware description language (HDL) model of system behavior at this point in the design process can greatly facilitate the rest of the design. The recommendation, of course, would be to use VHDL for a digital design. This standard language can then communicate the design through the subsequent steps in the design/development process.

It is also at this point in the hierarchical system design process where software and hardware implementation tradeoffs and system SW/HW partitioning are performed. It is here that there does not seem to be a well-defined common set of metrics that anchor this decision point; and subsequently these two processes (of software and hardware development) seem to be performed orthogonally, not linked in a manner that would help reduce design risk and improve design tracking. See Section 3.0 for recommendations in this area.

2.2.5.4 Hardware Design Partitioning—This step carries the conceptual design closer to a real hardware implementation. A conceptual design approach is selected and a hardware "design-to-baseline" is defined. This still does not establish the detailed design but establishes the point of embarkation for detailed design at the board, module, and IC levels. One also must recognize that at this point initial assumptions about board interconnection, i.e., backplane, are being introduced.

2.2.5.5 Outputs—F^3 software and hardware specifications for board (and potentially backplane), module, and IC designs. Simulation inputs and outputs that can be used as test excitations and responses for subsequent system-, board-, module-, and IC-levels of testing. A baseline system architecture.

2.2.5.6 Preliminary Design Review—Review the selected conceptual design. Compare the anticipated performance from the conceptual design with that sought by the performance specification. Check the completeness of the software and hardware F^3 specifications.

2.2.5.7 CAD Tools for High-Level Description Capture—There are several approaches to high-level system modeling and each has its merits. The design itself strongly influences the choice of a preferred approach. Figure 2.2.5.7-1 suggests a concept for a system simulation/emulation integrated product development testbed. The need for the design/modeling flexibility implied by such a testbed follows from the many design issues that must be addressed at this critical juncture in the design process. As shown previously in Figure 2.2-2, all of the parallel design/development paths (for software, board-level, module-level, and ASIC-level) flow out of this step in the overall process.

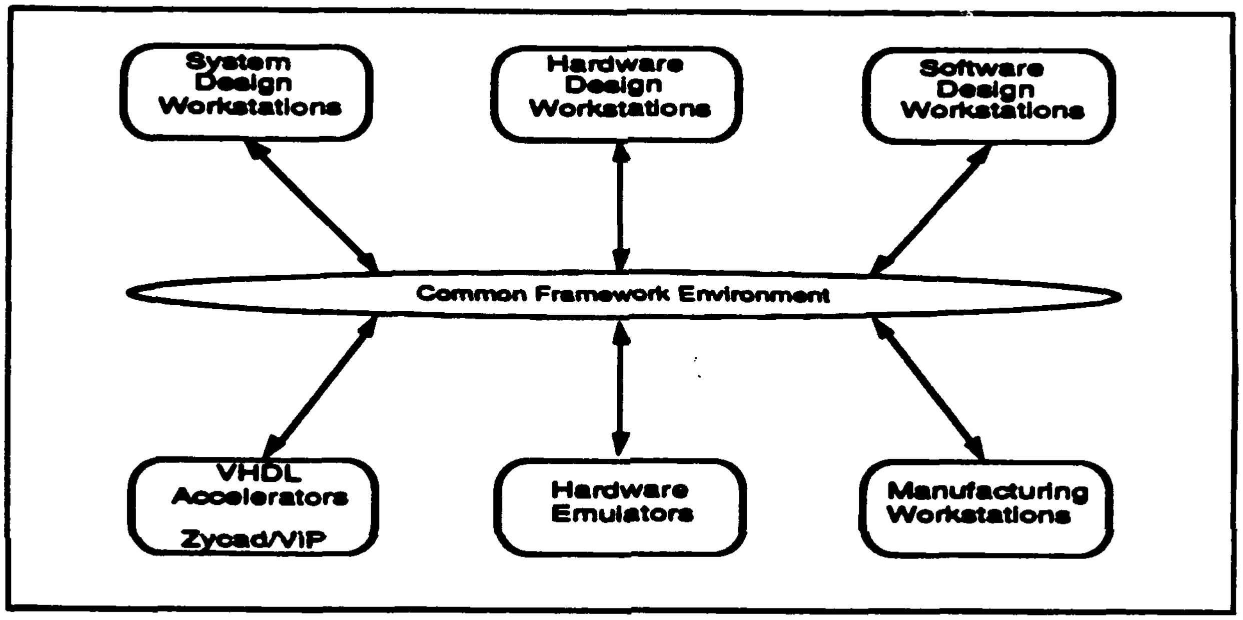

System Design Workstations
Hardware Design Workstations
Software Design Workstations
Common Framework Environment
VHDL Accelerators Zycad/VIP
Hardware Emulators
Manufacturing Workstations

Figure 2.2.5.7-1. System Simulation/Emulation Product Development Testbed

CAD tools such as Mentor's System Design Station (SDS), SES/workbench, and I-Logic's Statemate provide the capability for high-level behavioral modeling of a system. SDS and Statemate animate the system's behavior so one can step through the system's logic sequence and observe data flows and node firings. Node behavior can be modeled for either software or hardware implementations so the tools for performing software versus hardware implementation tradeoffs are becoming available. SES/workbench interfaces with a software development tool called Software Through Pictures that can then carry the design process into the software development domain.

2.2.5.8 Issues/Risks—We have stated that the high-level description capture task is a critical step in the product development process because it is the step that the board, module, ASIC, and software developments emerge. The success or failure of RASSP design methodology is probably tied to this point in the design process. It is then very important that the design methodology and associated CAD, CAM, and CAT tools share as much common ground as possible at this point, perhaps via a shared framework environment as implied in the center of Figure 2.2.5-7-1. What will it take to make this successful? A number of suggestions follow:

1. Urge CFI to coordinate EDA vendors cooperation and movement in this direction.
2. Encourage industry acceptance of form, fit, and function as specific domains for design description and documentation.
3. Establish conventions for design documentation, preferably maximizing the use of a common hardware description language such as VHDL.
4. Encourage development of reusable and accessible model libraries for standard parts and with standard metrics to be compatible with the tool types shown in Figure 2.2.5.7-1.
5. Develop common high-level representation and metrics to secure the link between CASE and CAD tools.
6. Make sure that CAM and CAT are part of this common design domain.

Table 2.2.5.8-1 summarizes the current status of many of these system design activities.

Table 2.2.5.8-1. System Design Status Summary

Technology	*SOA*	*Programs*	*Standards*	*Issues*	*Solutions*
System Architecture					
• System Analysis	SES Workbench ADAS, Statemate, Ascent Logic RDD, SPW, Synopsys VHDL System Simulator; IR&D	GPD/CDG	VHDL, CFI	Tools poorly integrated, narrow focus	EDA vendors move towards CFI
• Documentation	2167A requirements generated from CASE tools	CEENSS	2167A	Current standards hard to integrate with new design process	Establish/change new documentation requirements
• Requirements Tracking	Ascent Logic RDD, Honeywell ARRTS	CEENSS		Tools tend to be stand-alone; little formalism	Establish conventions in VHDL/documentations
• Trade Off Analysis	Internal tools; Honeywell's ATeT, Mentor/TI DFM, MSDA, ADAS	ASEM	VHDL	No conventional metrics, representations, or libraries	Establish reuse libraries with standard metrics; integrated trade off tools
• Formal Verification	CLSI Parts, Vertex Boolean Verifier, IR&D		VHDL	Still very immature; existing tools have limited focus	Standard subset; domain specific verification
• System Level Modeling/Simulation	VHDL models, Acceleration (Zycad, IKOS), GLSS	F-22 Protocol, GPD/CDG	VHDL, IEEE 1164	Speed, speed and speed; lack of modeling standards; poor analog/digital simulators	"System Modeling Handbook," behavioral accelerators; develop a/d simulators
• System Hardware/ Software Partitioning	System analysis tools; manual	CEENSS	VHDL, Ada	No good link between CASE and EDA tools	Develop common high level representation
• System Test Generation	System Test Synthesis, Synopsys Test Compiler	ASEM	IEEE 1149	Tests developed from relatively low level (RTL) description	Test development from requirements, test decomposition/tracking

ADAS	Architecture Design and Assessment System	MSDA	Multichip System Design Advisor
RDD	Requirements Driven Design	DFM	Design for Manufacturability
ARRTS	Avionics Requirements to Test Tracking System	GLSS	Gate Level System Simulation
CEENSS	Continuous Electronic Enhancement using Simulatable Specifications		
ASEM	Application Specific Electronic Module		

If one assumes a 10 to 15 year product life cycle with model year upgrades, then the product development environment/testbed/data base and associated EDA tools must evolve with the product. This will require adequate support personnel, planned tool upgrades, and training. Tool interoperability standards and non-proprietary modeling techniques must be established to reduce the risks associated with a tool vendor going out of business.

Finally, educational system course work needs to be developed that focuses upon top-down design and all the real world implementation issues/tradeoffs that must be addressed at this critical juncture in the product development process. This course work must include a broad overview to CAD, CAM, CAT, and CASE.

2.2.5.9 System Architecture for RASSP—A baseline system architecture will be one of the important outputs from the high-level description capture task. Honeywell has proposed utilizing the RASSP architecture both for front-end sensor processing and also for back-end display processing thus broadening its applicability and potential product volume. This concept was mentioned previously in Figure 2.2.3.1-1. We emphasize here that each element and interface shown in the referenced figure ultimately will be required to have its own F^3 specification.

The success of RASSP will depend to a large extent on how flexible and expandable the system architecture is. The RASSP architecture needs to support the following features:

- Dual use parts (commercial and military);
- Many model year upgrade paths;
- Upgrades to new capabilities;
- Maximize the processing power.

The basic approaches to achieving these architecture goals are: (1) to use new technologies for high-calculation and high-interconnect throughput and (2) to use open, standardized interfaces (see Figure 2.2.3.1-1).

The high speed signal processing throughput is achievable through new IC technologies, e.g., GaAs, submicron Si, and through new interconnect technologies. The high throughput interconnects are achievable with photonics/optical technology, as shown in Figure 2.2.5.9-1, and with multichip modules (MCMs). Under the RASSP program, these two technologies have to be combined together as shown in Figure 2.2.5.9-2. Both intra- and inter-MCM connections are implemented optically with polyimide waveguides and/or optical fiber. Optical interconnects will solve the backplane bandwidth density and power dissipation problems. In addition, the photonic interconnects are secure and electromagnetic influence (EMI) immune (see Figure 2.2.5.93).

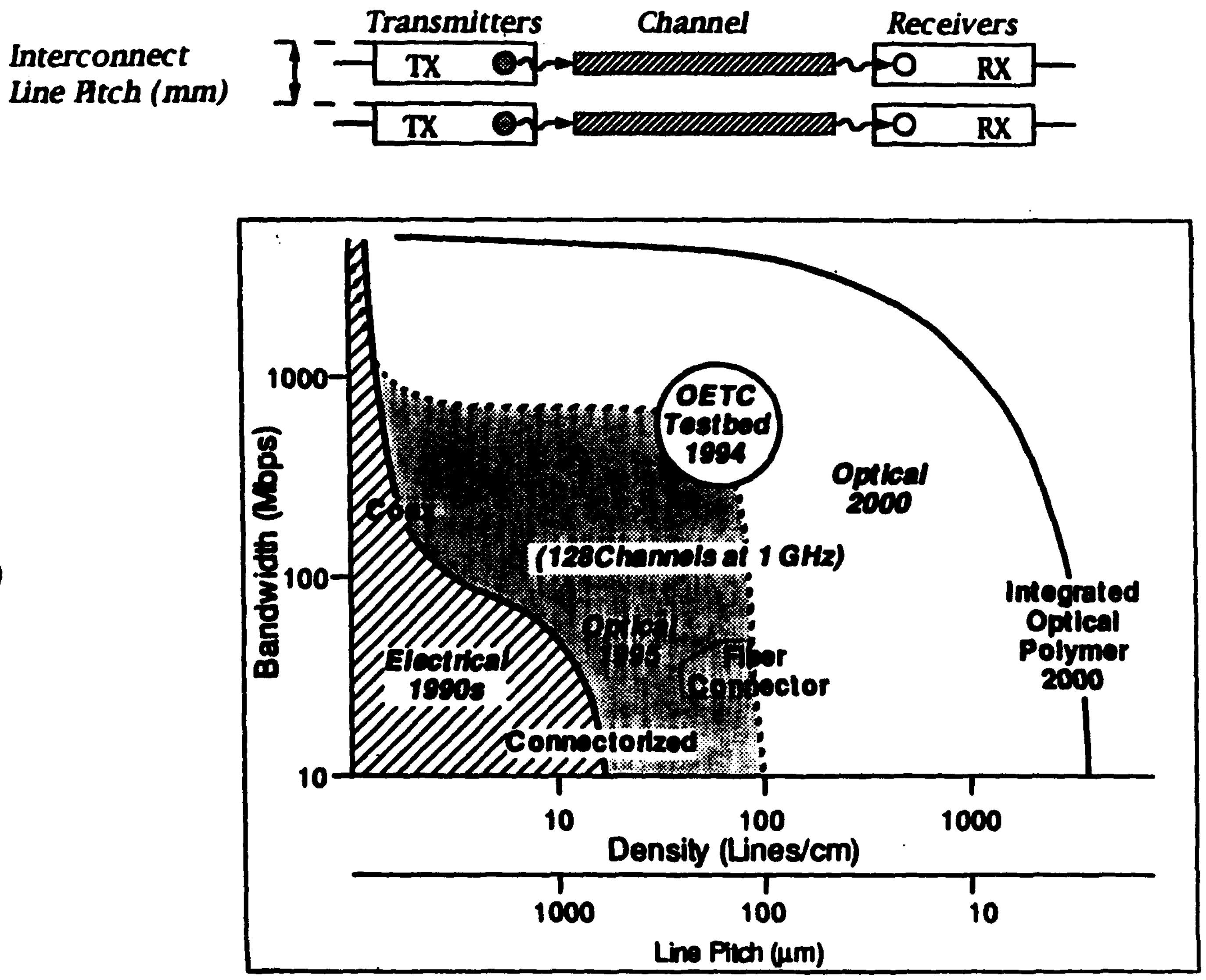


Figure 2.2.5.9-1. Interconnect Technology Comparisons

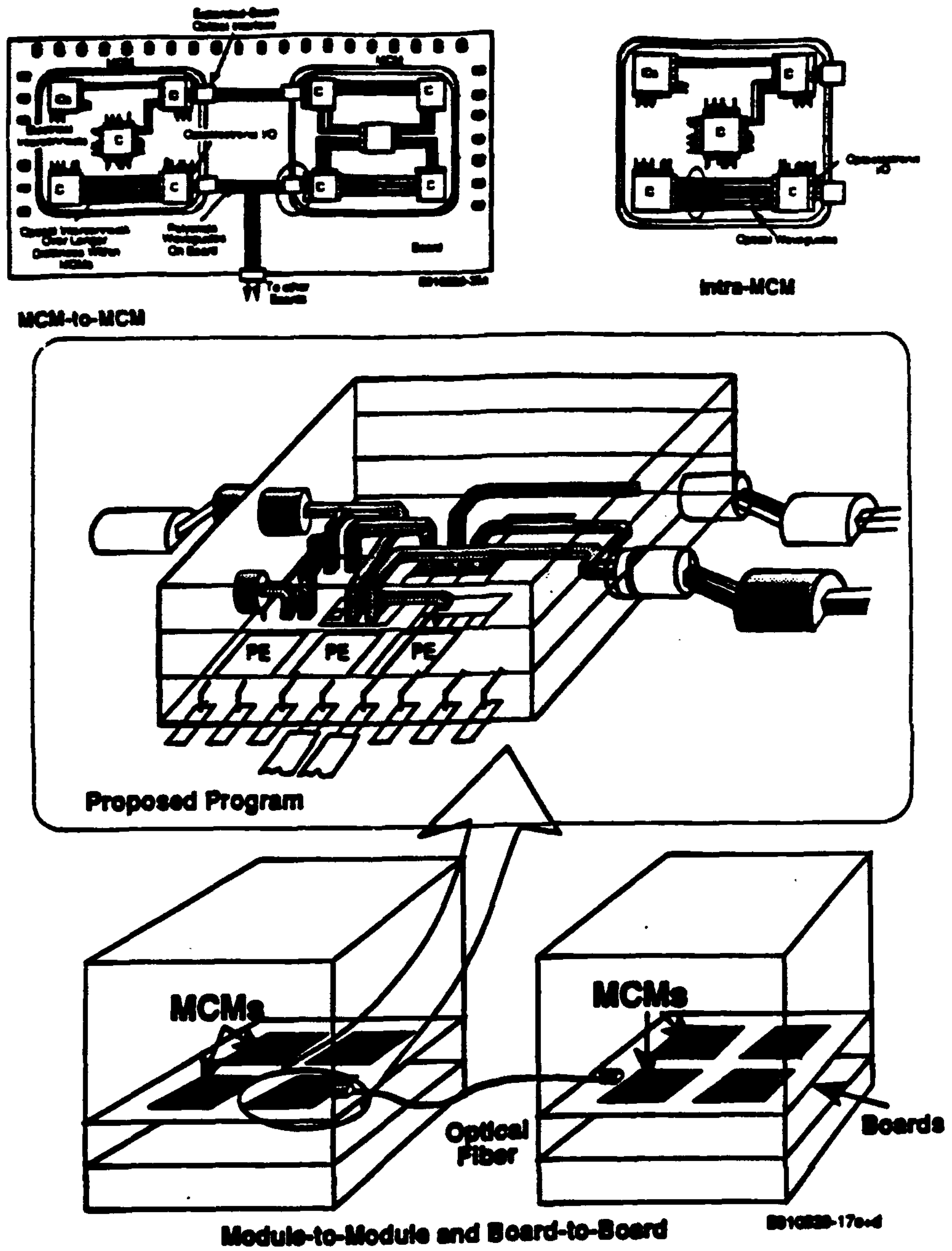


Figure 2.2.5.9-2. Optical Interconnect Technology

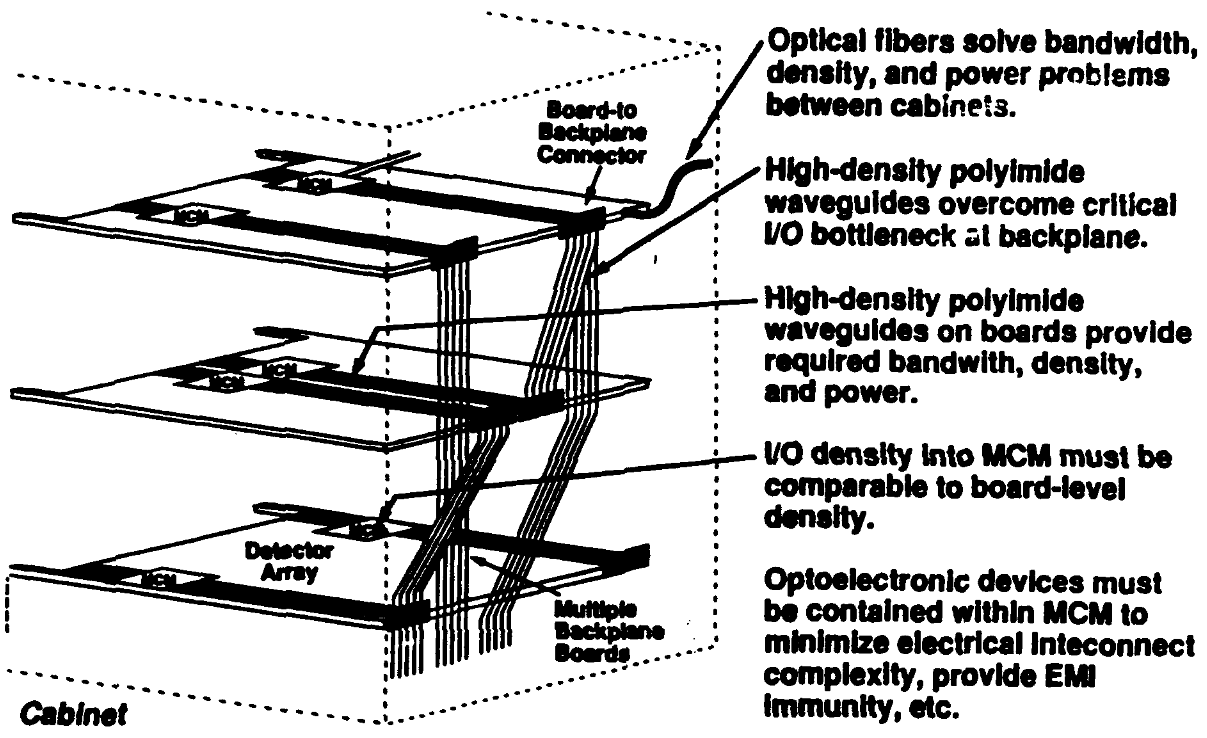

Optical fibers solve bandwidth, density, and power problems between cabinets.
High-density polyimide waveguides overcome critical I/O bottleneck at backplane.
High-density polyimide waveguides on boards provide required bandwith, density, and power.
I/O density into MCM must be comparable to board-level density.
Optoelectronic devices must be contained within MCM to minimize electrical inteconnect complexity, provide EMI immunity, etc.
Board-to Backplane Connector
MCM
Detector Array
Multiple Backplane Boards
Cabinet

Figure 2.2.5.9-3. Modification for Optical Interconnects in MCMs

2.3 Hardware Design

Hardware design commences from the F^3 board, module, and IC specs and can proceed in parallel design flows as shown in Figure 2.2-2. Board and module design flows involve similar steps, although not necessarily the same tools, even though the physical implementation of the board and module is different. For example, the module uses bare die ICs versus packaged ICs for board assembly. However, if the bare die are characterized similarly to packaged die, the board and module assembly stays the same. The scale and materials used for boards and MCMs are also different, but once the models for thermal and electromagnetic modeling are set up correctly, the design steps for boards and modules are the same. The design steps are shown in Figure 2.3-1. The top-level detailed ASIC design is previously shown in Figure 2.2.2-2.

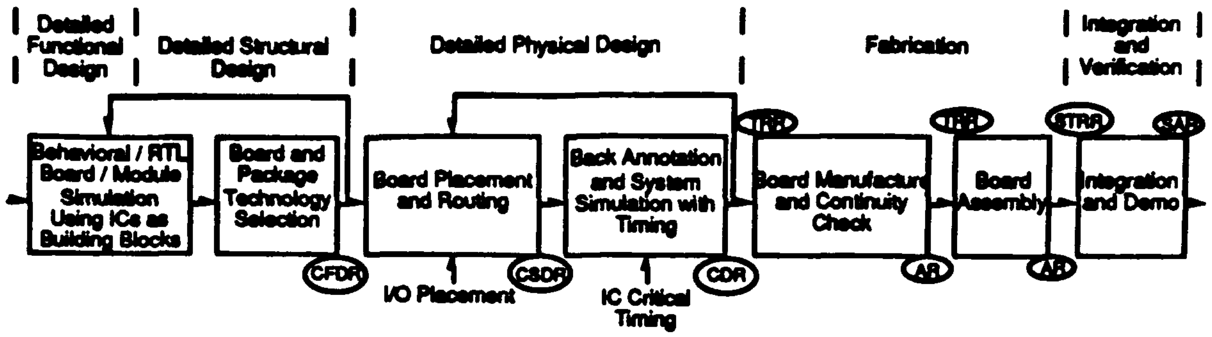

Detailed Functional Design
Detailed Structural Design
Detailed Physical Design
Fabrication
Integration and Verification
Behavioral / RTL Board / Module Simulation Using ICs as Building Blocks
Board and Package Technology Selection
Board Placement and Routing
Back Annotation and System Simulation with Timing
Board Manufacture and Continuity Check
Board Assembly
Integration and Demo
TRR
TRR
STRR
SAR
CFDR
CSDR
CDR
AR
AR
I/O Placement
IC Critical Timing

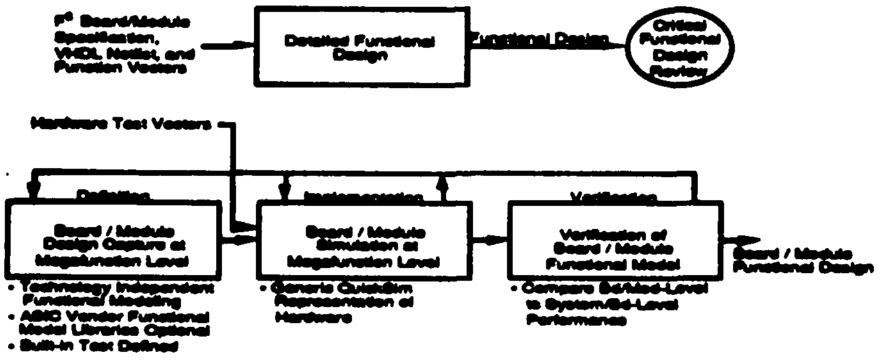
Figure 2.3-1. Board and Module Design Flows

Many hardware design tools exist that allow a designer to utilize graphical and behavioral models of the system for designing to meet that system's performance requirements at a detailed level. These tools fall into two general classes, hardware-level CAD and computer-aided prototyping. Examples of these tools include:

- Schematic Capture
- Timing Design
- ASIC Design
  - Design/Simulation
  - Synthesizers
  - Test Compilers
  - Automatic Test Pattern Generators
- PLD/FPGA Design
- PCB Design and Layout
- MCM Design and Layout
- Design/Documentation

2.3.1 Detailed Board/Module Functional Simulation

This phase of the design process will perform the detailed implementation of the board or module functional specification.


Board/Module Specification, VHDL Netlist, and Function Vectors
Detailed Functional Design
Functional Design
Critical Functional Design Review
Hardware Test Vectors
Definition
Implementation
Verification
Board / Module Design Capture at Megafunction Level
Board / Module Simulation at Megafunction Level
Verification of Board / Module Functional Model
Board / Module Functional Design
• Technology Independent Functional Modeling
• ASIC Vendor Functional Model Libraries Optional
• Built-in Test Defined
• Generic QuickSim Representation of Hardware
• Compare Bd/Mod-Level to System/Bd-Level Performance

Sample CAD Tools, Mentor Board Station/ Mentor MCM Station
Vantage VHDL.Synopsys. Dazix, Cadence, OrCAD. Viewlogic. Test Editor
Figure 2.3.1-1. Detailed Functional Design

2.3.1.1 Inputs—The functional description from the F^3 Preliminary Design Specification. The inputs may also include simulator netlists and test vectors depending on the CAD tools used in the previous design phase.

2.3.1.2 Board/Module Design Capture at Megafunction Level—Each board or modul_ is modeled as consisting of its major blocks, i.e., megafunctions. This modeling is technology independent but still can be done using ASIC vendor functional models/libraries. The modeling language is VHDL (preferred). The model representation can be either behavioral or RTL or a combination of each, and includes the test methodology implemented at the board or module level; e.g., boundary-scan blocks and interconnects. The basic design capture process captures the design as either a schematic, netlist, or both. Figure 2.3.1.2-1 below shows an example of how the tools of different vendors can be integrated. At the design capture level one must allow for different alternatives. For example, a text editor is included for plain old ASCI inputs or outputs. For simple (not highly complex) boards, a tool such as OrCAD operating on a PC can work fine. OrCAD is an interesting example because it supports popular output formats (Gerber, Postscript, and Calcomp) and provides low-cost access to vendors and to design documentation. Gerber is a popular format for interfacing to PCB manufacturers.

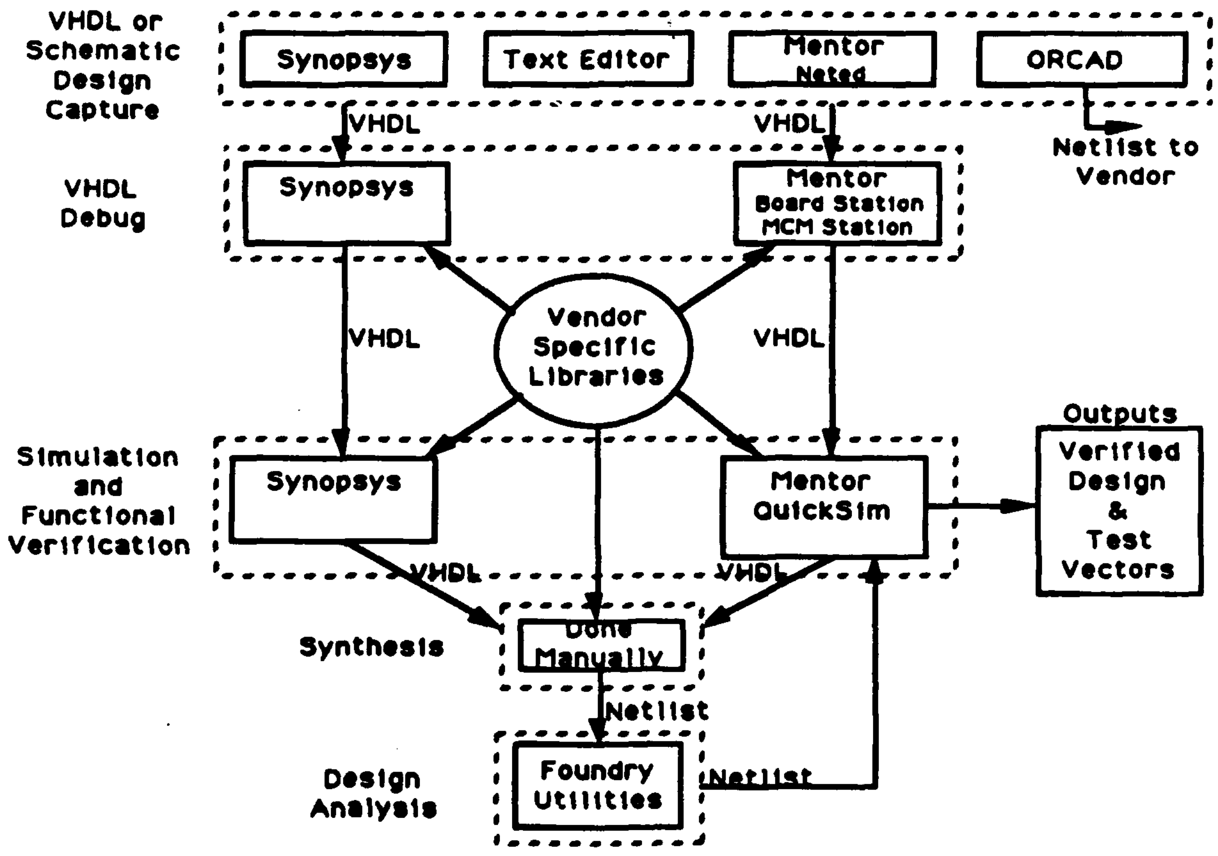


Figure 2.3.1.2-1. Example of Integration of Different Vendor Tools Emphasizing Use of VHDL

2.3.1.3 Board/Module Simulation at Megafunction Level—There are many CAD tools for performing behavioral and/or RTL-level simulation. Figure 2.3.1.2-1 also includes Mentor and Synopsys CAD tools but could equally have included DAZIX, Cadence, etc. The important missing element in the referenced figure is synthesis. For both board and module design, this step is basically performed manually. At the board level the designer is selecting standard, hopefully available, parts that perform the required megafunctions. Designer experience and the decision options available cannot be practically captured by an automatic tool. Perhaps an interactive AI tool could be developed as a knowledge-based design guide, but the technology moves too rapidly to keep such a tool up to date. "Seamless" design for board-level synthesis may simply mean that the designer is in the loop.

A similar situation occurs for design synthesis at the module level. Here the designer is mainly dealing with standard, hopefully available, chips. The decision alternatives are again too numerous to be automated.

2.3.1.4 Verification of Board/Module Functional Model—The simulated functional performance will either be compared with the F^3 board/module performance specification or with the test vectors input from the previous design phase.

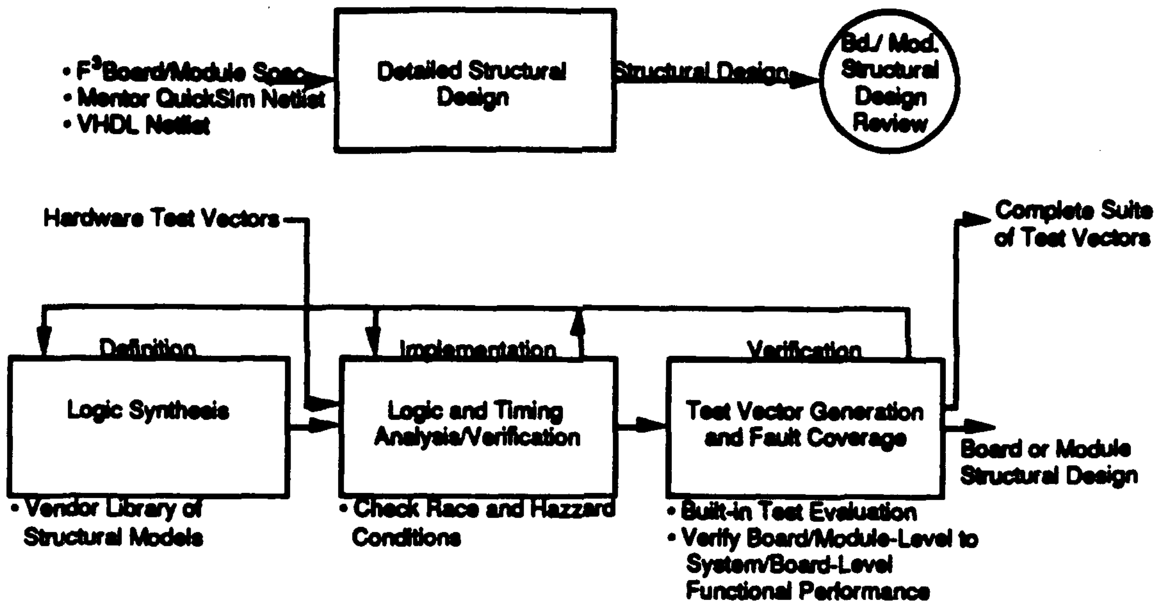
2.3.1.5 Outputs—Board/module functional design to the major block level.

2.3.1.6 Critical Functional Design Review—This is the first of three critical design reviews, this one focused on board or module functionality.Before proceeding to structural design, acceptability of the functional performance will be reviewed.

2.3.1.7 CAD Tools—As stated above, there are many CAD tools available to accomplish the detailed board-level and module-level functional design. One must look at detailed tool features to differentiate one tool from another. The key to the environment, as shown in Figure 2.3.1.2-1 and to achieve the seamless design environment, including interoperable tools, sought by the RASSP program, is to require standard tool interfaces. This is the goal of the CAD Framework Initiative (CFI), and through EDA industry participation and cooperation, this goal can be achieved. During the RASSP implementation phase, two-way communication and technical interaction should be maintained with CFI.

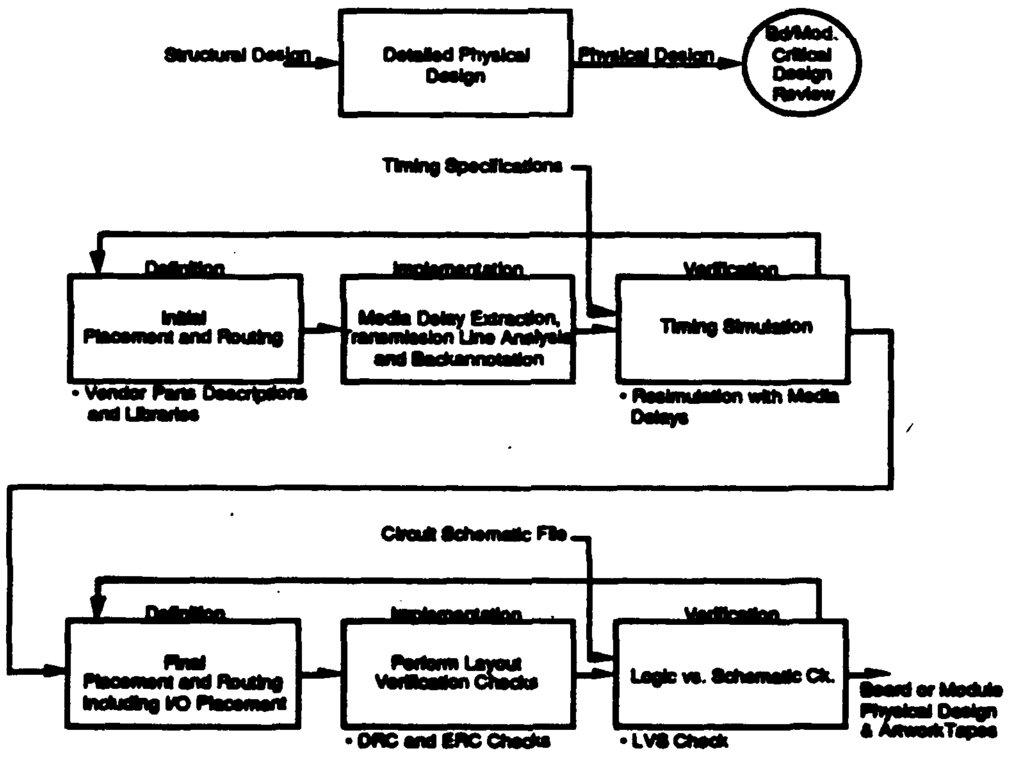
2.3.2 Detailed Structural Design

This step in the board or module design process will perform the detailed implementation of the board or module structural specification. Since our goal here has been to merely demonstrate a canonical formalism for implementing the product development process, we will abbreviate the discussion by subsequently including only the major steps and their diagrams.


• F^{3}Board/Module Spec
• Mentor QuickSim Netlist
• VHDL Netlist
Detailed Structural Design
Structural Design
Bd./ Mod. Structural Design Review
Hardware Test Vectors
Complete Suite of Test Vectors
Definition
Implementation
Verification
Logic Synthesis
Logic and Timing Analysis/Verification
Test Vector Generation and Fault Coverage
Board or Module Structural Design
• Vendor Library of Structural Models
• Check Race and Hazzard Conditions
• Built-in Test Evaluation
• Verify Board/Module-Level to System/Board-Level Functional Performance

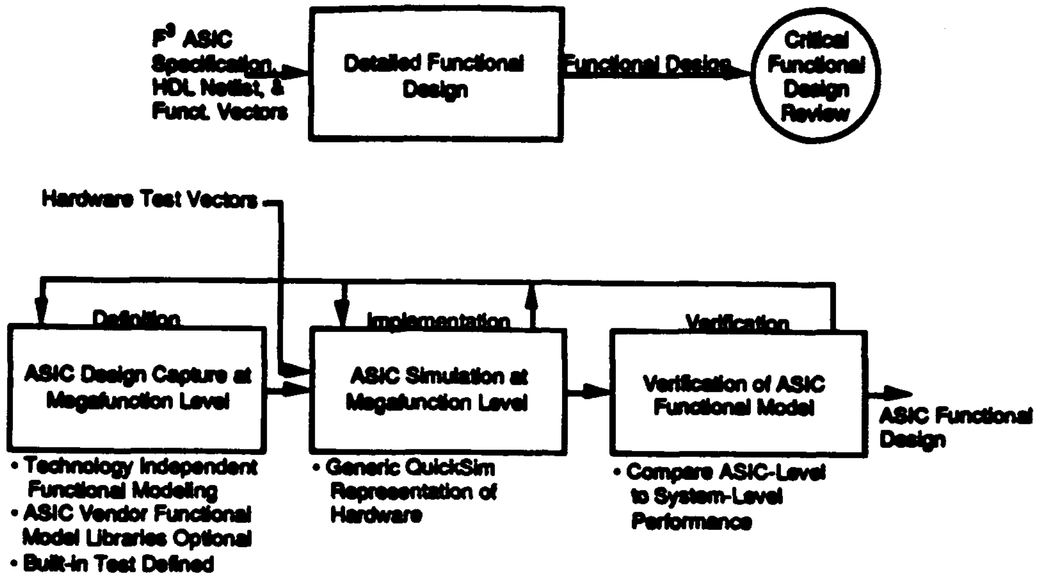
Sample CAD Tools, Mentor Board Station/Mentor MCM Station, Vantage VHDL, Synopsys. Dazix, Cadence
Figure 2.3.2-1. Board Detailed Structural Design

2.3.3 Detailed Physical Design


Structural Design
Detailed Physical Design
Physical Design
Bd/Mod. Critical Design Review
Timing Specifications
Definition
Implementation
Verification
Initial Placement and Routing
Media Delay Extraction, Transmission Line Analysis and Backannotation
Timing Simulation
• Vendor Parts Descriptions and Libraries
• Resimulation with Media Delays
Circuit Schematic File
Definition
Implementation
Verification
Final Placement and Routing Including I/O Placement
Perform Layout Verification Checks
Logic vs. Schematic Ck.
Board or Module Physical Design & ArtworkTapes
• DRC and ERC Checks
• LVS Check

Sample CAD Tools, Mentor Board Station/Mentor MCM Station, Dazix, Cadence, OrCAD, EEsof
Figure 2.3.3-1. Detailed Board Physical Design

2.3.4 Detailed ASIC Functional Design


F³ ASIC Specification, HDL Netlist, & Funct. Vectors
Detailed Functional Design
Functional Design
Critical Functional Design Review
Hardware Test Vectors
Definition
Implementation
Verification
ASIC Design Capture at Megafunction Level
ASIC Simulation at Megafunction Level
Verification of ASIC Functional Model
ASIC Functional Design
• Technology Independent Functional Modeling
• ASIC Vendor Functional Model Libraries Optional
• Built-in Test Defined
• Generic QuickSim Representation of Hardware
• Compare ASIC-Level to System-Level Performance

Sample CAD Tools, Enhanced VHDL. QuickSim. Galahad
Figure 2.3.4-1. Detailed ASIC Functional Design

2.3.4.1 CAD Tools

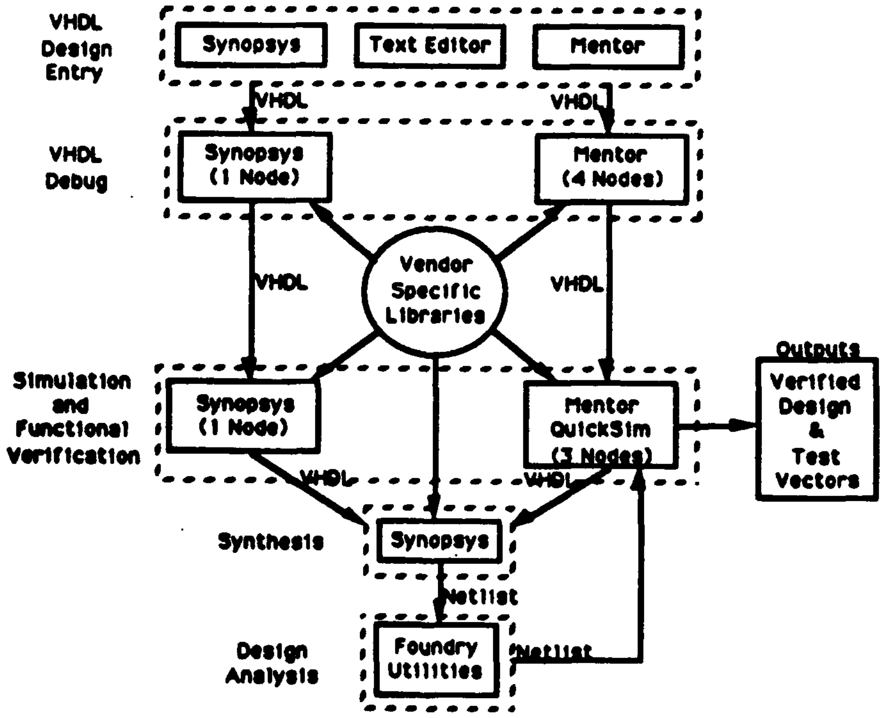
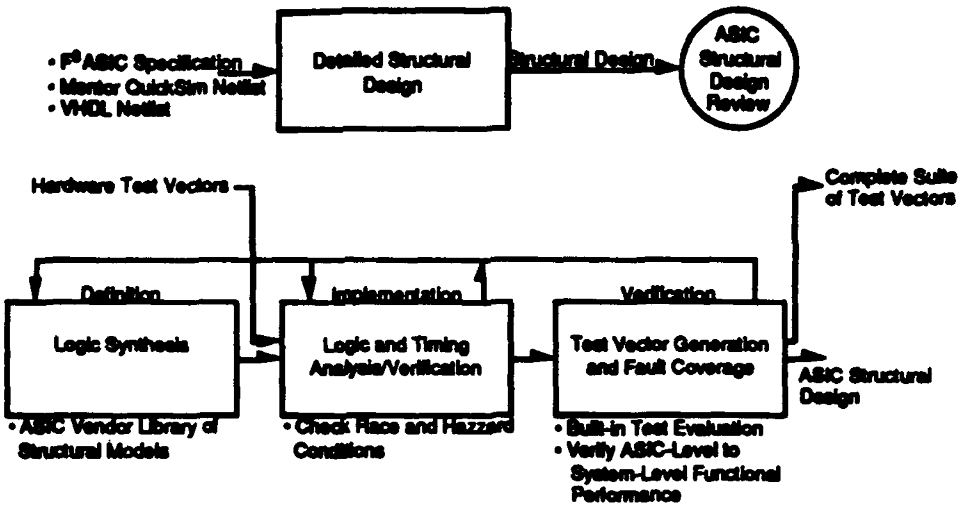


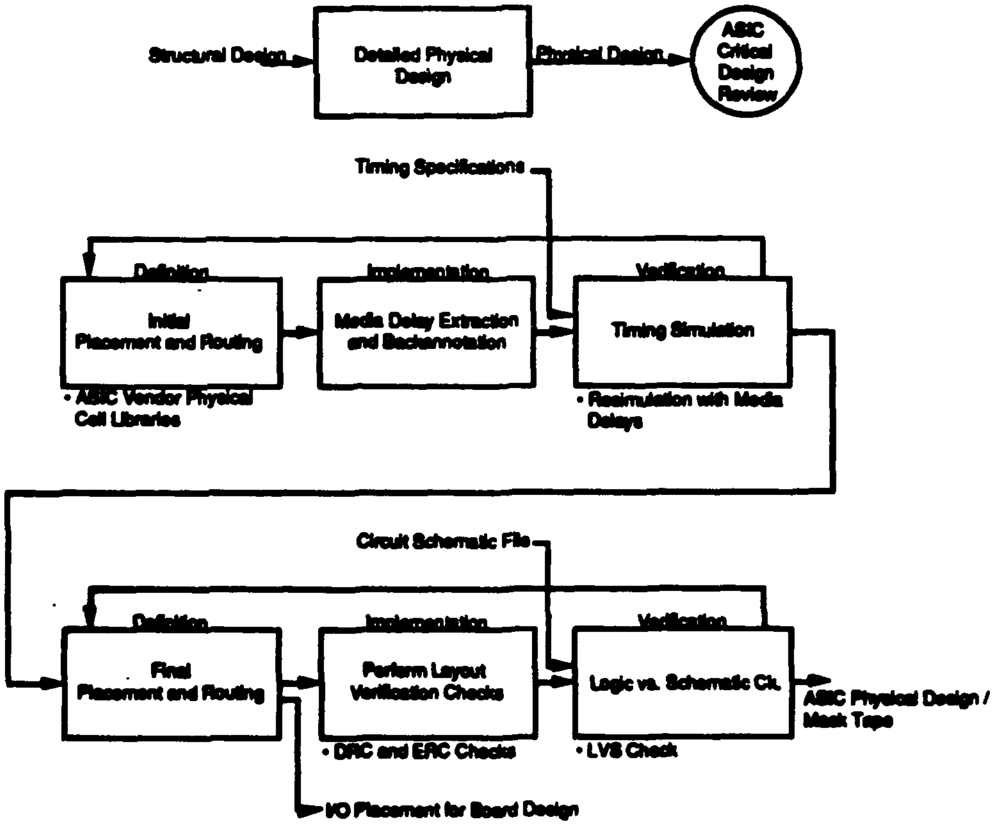
Figure 2.3.4.1-1. Current VHDL Design Flow for Detailed ASIC Functional and Structural Design

2.3.5 Detailed Structural Design


• F³ ASIC Specification
• Mentor QuickSim Netlist
• VHDL Netlist
Detailed Structural Design
Structural Design
ASIC Structural Design Review
Hardware Test Vectors
Complete Suite of Test Vectors
Definition
Implementation
Verification
Logic Synthesis
Logic and Timing Analysis/Verification
Test Vector Generation and Fault Coverage
ASIC Structural Design
• ASIC Vendor Library of Structural Models
• Check Race and Hazzard Conditions
• Built-in Test Evaluation
• Verify ASIC-Level to System-Level Functional Performance

Sample CAD Tools, Enhanced VHDL, Synopsys. QuickSim
Figure 2.3.5-1. Detailed ASIC Structural Design

2.3.6 Detailed Physical Design


Structural Design
Detailed Physical Design
Physical Design
ASIC Critical Design Review
Timing Specifications
Definition
Implementation
Verification
Initial Placement and Routing
Media Delay Extraction and Backannotation
Timing Simulation
• ASIC Vendor Physical Cell Libraries
• Resimulation with Media Delays
Circuit Schematic File
Definition
Implementation
Verification
Final Placement and Routing
Perform Layout Verification Checks
Logic vs. Schematic Ck.
ASIC Physical Design / Mask Tape
• DRC and ERC Checks
• LVS Check
I/O Placement for Board Design

Sample CAD Tools, Mentor "Layout" Stations, Back Annotation, QuickSim, Dracula or Checkmate
Figure 2.3.6-1. Detailed Physical Design

2.3.7 Model Library Development

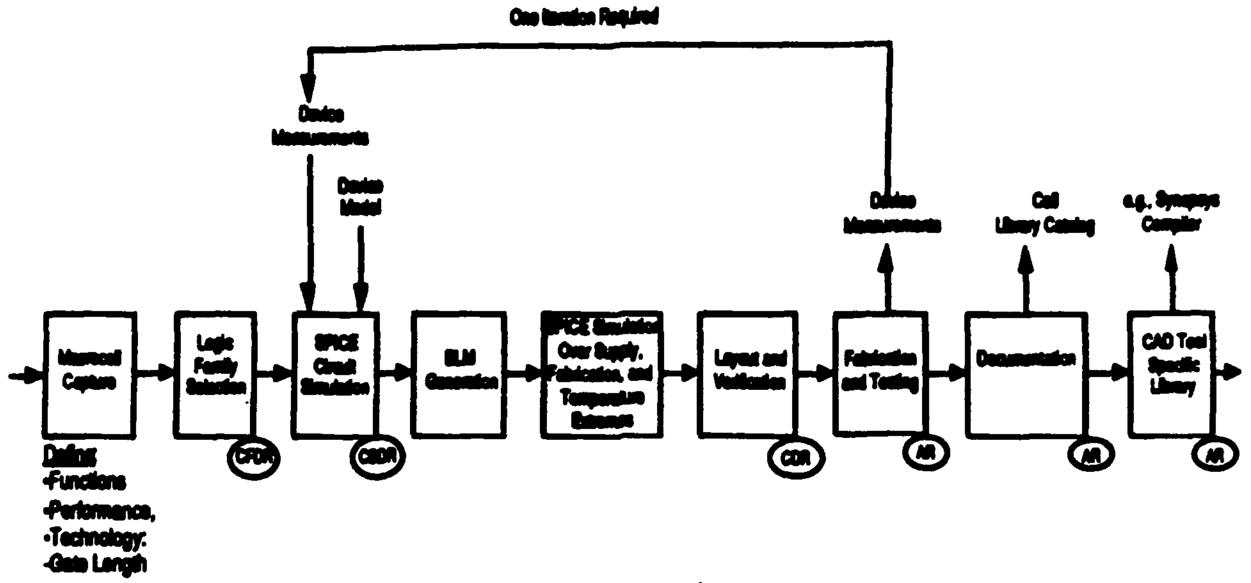


Figure 2.3.7-1. Model Library Development Process

2.3.8 Fabrication, Test, and Packaging

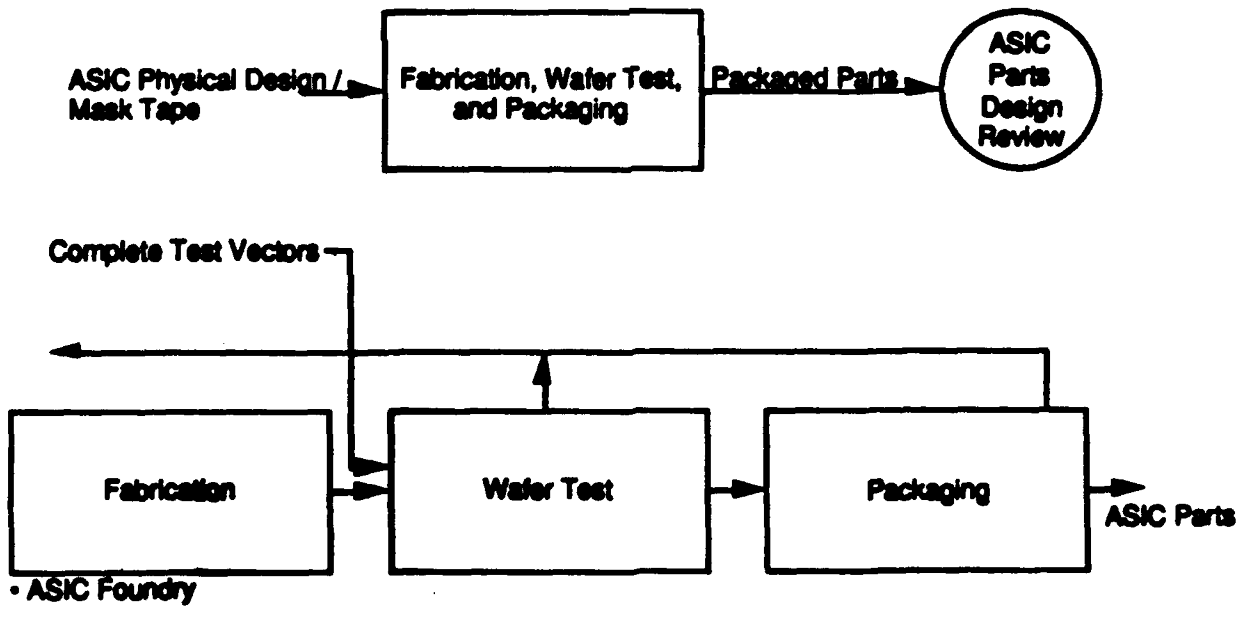


Figure 2.3.8-1. Fabrication, Wafer Test, and Packaging

2.3.9 Testbed or System Integration

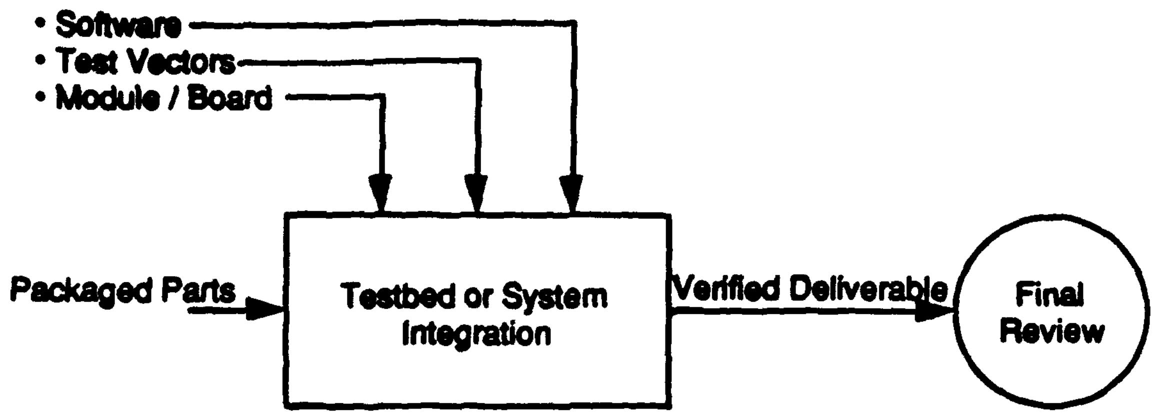


Figure 2.3.9-1. Testbed or System Integration

2.3.10 Design Reviews

- Project Plan Review (PPR)
- Systems Requirements Review (SRR)
- Performance Spec. Review (PSR)
- Preliminary Design Review (PDR)
- Critical Functional Design Review (CFDR)
- Critical Structural Design Review (CSDR)
- Critical Design Review (CDR)
- Test Readiness Review (TRR)
- Acceptance Review (AR)
- System Test Readiness Review (STRR)
- System Acceptance Review (SAR)

2.3.11 Documentation

- System Requirements Document (SRD)
- System Performance Spec. (SPS)
- HDL Documentation
- ASIC Documentation
- Board/Module Documentation
- Cell Library Documentation
- CAD Tool Specific Library Documentation

2.4 Software Design

The primary issue related to software design is its role with respect to system analysis and hardware design. The software design tasks and data flow is shown in Figure 2.4-1. RASSP applications stress all of these steps due to their demand for high performance and reliability.

An example of a first step toward uniting these development areas for embedded systems can be seen in Integrated Design Automation System (IDAS), a product of JRS Research Laboratories. The IDAS toolset derives hardware and software design information, maps Ada programs onto machines described in VHDL, synthesizes machines described in VHDL from Ada-coded specifications, retargets microcode compiler tools from VHDL machine descriptions, and assists in the quantitative evaluation of design alternatives. To make systems such as IDAS more effective for RASSP software development, the following technology areas must progress:

- Domain-specific software architecture and support
- V and V/test
- Programming languages
- Embedded operating systems
- Micro-code development

The state-of-the art programs, standards, issues, and RASSP recommendations are presented in Table 2.4-1 and described in the following subsections.

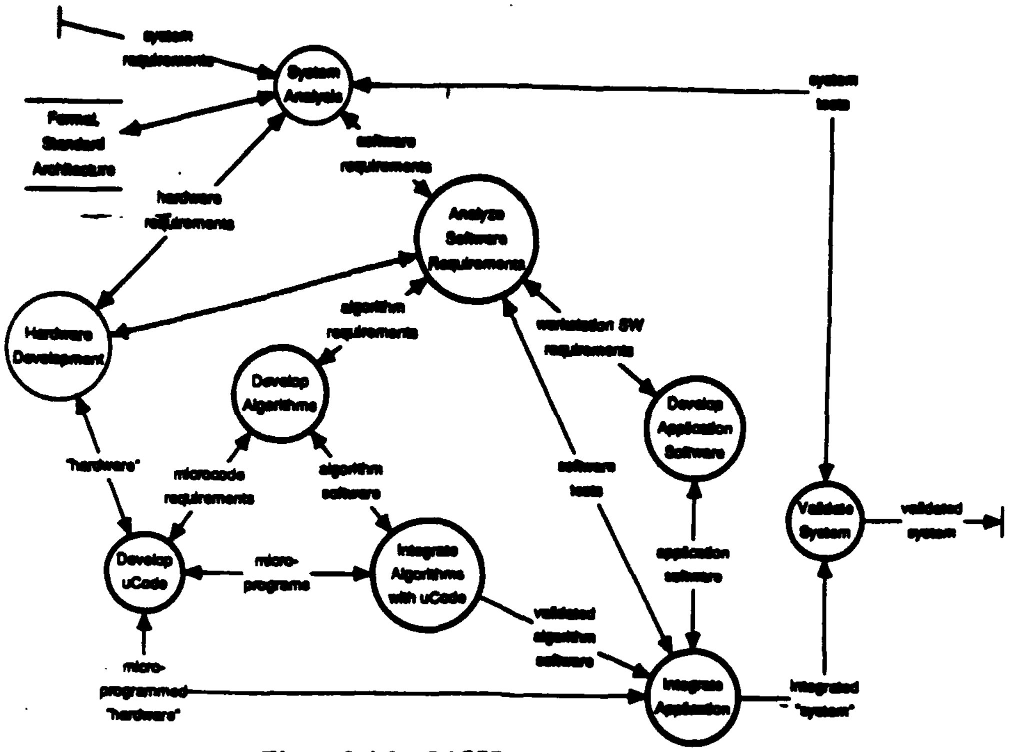


Figure 2.4-1. RASSP Software Development

Table 2.4-1. Software Development Issues

Technology	SOA	Pgms	Stds	Issues	RASSP Recommendations
Software Arch.					
• Domain Analysis	IDEF, OOA	SEI, DSSA	none	Little automation to support capture and organization of domain model	Use "smart" tools wherever possible to permit upgrade
• Architecture Specification	Dataflow, block diagram, MILs (IDL, Polylith, APPSL)	DSSA, Proto-Tech	none	Domain-specificity of arch. specs limit tool support	Establish arch. spec language for SP
• Analysis/Simulation	Generic: Petri net, queuing, state-transition, etc.; Special purpose		VHDL	SW complexity quickly saturates generic tools	Employ composable, hierarchical models
Domain Specific Tools	Khoros, KB-Vision, Image Flow; Adapt, Apply, ASSIGN			Low support for specific target architectures; no tools to analyze target system performance	Develop versions targeted towards RASSP architecture.
V&V/Test	Semi-automatic test generation (T, SofTest), symbolic debuggers	ASID/VS		loose integration with simulation-oriented environment; weak traceability to design rationale	Pursue domain-specific test generation
Programming Languages	General-purpose, structured, object-oriented languages; front-end CASE tools (teamwork®, StP®, Rational)		Ada (9x), C, C++, Lisp	Language subsets generally needed for robust, real-time, secure applications. Interoperability.	Standardize language subsets
Embedded operating systems	Mach, Alpha, RT-Unix, proprietary	CHPC, ISIS	POSIX	Fault-tolerance still in basic research stage	Accelerate research in loosely-coupled F-T, strengthen focus on SP
Micro-code development	Microcode compilation, semi-automatic retargeting of compiler	IDAS	None	Microcode compilation yields code slightly less compact than human-developed	Aim μCode compiler (retargeting) technology to standard SP architecture

2.4.1 Domain-Specific Software Architecture and Support

For the purposes of this study, domain-specific software architecture and support was further decomposed into three technology areas; domain analysis, architecture specification, and domain-specific tools.

1) Domain Analysis—The ultimate objective of domain analysis is rapid, high-quality, continuously improved, product development. These are clearly valuable attributes for RASSP. Up to this point, domain analysis as a discipline has been emerging primarily from the software research community and can be characterized as:

> an activity occurring prior to systems analysis and whose output (i.e., a domain model) supports systems analysis in the same way that systems analysis output (i.e., requirements analysis and specifications document) supports the systems designer's tasks... In DA we try to generalize all systems in an application domain by means of a domain model that transcends specific applications. The next step is to define a domain specific language. [Prieto-Diaz87]

Traditionally, the tendency is to treat software and hardware development processes independently. Domain modeling is an important step towards joining them in a productive way.

Domain analysis methods—most often based on a form of object-oriented analysis—are relatively new, and as a result, few organizations have developed significant domain models. Fewer still are commercial tools that provide robust support for the methods. Information modeling tools, such as those based on the IDEF (ICAM (Integrated Computer-Aided Manufacturing) DEFinition) methodology lack the ability to describe system behavior. The emerging object-oriented analysis tools (e.g., Peter Coad's Object Designer) come closer to providing the needed expressiveness, but lack features necessary for integrating the wide variety of information sources that domain modeling depends upon.

2) Software Architecture Specification—Several languages, both textual and graphical, exist for the specification of software architecture (e.g., flow graph, block diagram, Petri net, Module Interconnection Languages (MILs). Practically none of these, however, are specific to the domain of signal processing. This is an area of critical need, for without application-specific architecture specification languages, there is little basis for significantly improving the efficiency of quality software development for signal processing.

Some promising research is presently underway at CMU, where the Adapt language is being developed [WEBB92]. This work has also defined a layering of architectural concerns that provide a framework for future specification language development.

The DARPA Domain-Specific Software Architectures (DSSA) program is defining several such architecture specification formalisms for a variety of RASSP-relevant domains, including vehicle control, vetronics, C3I and sensor processing. An example of software architecture is shown in Figure 2.4.1-1.

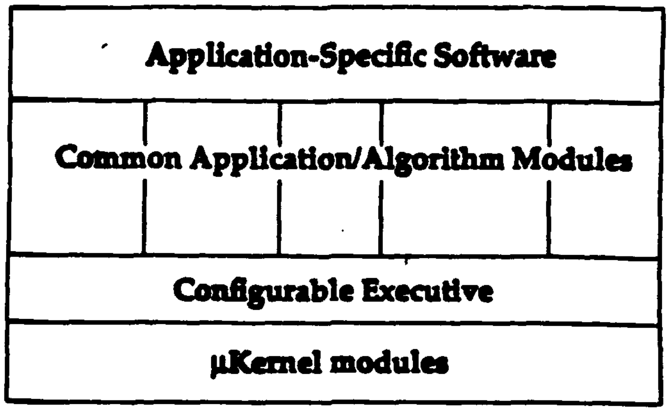

Application-Specific Software
Common Application/Algorithm Modules
Configurable Executive
μKernel modules

Figure 2.4.1-1. RASSP Software Architecture

3) Domain-Specific Tools—Several tools are available for various aspects of signal processing, e.g., [Khoros]. Domain-specific tools should be based on a common architecture specification formalism that integrates all aspects of the RASSP architecture.

   The DARPA DSSA program is developing and integrating domain-specific toolsets in each of its target domains, as shown in Figure 2.4.1-2.

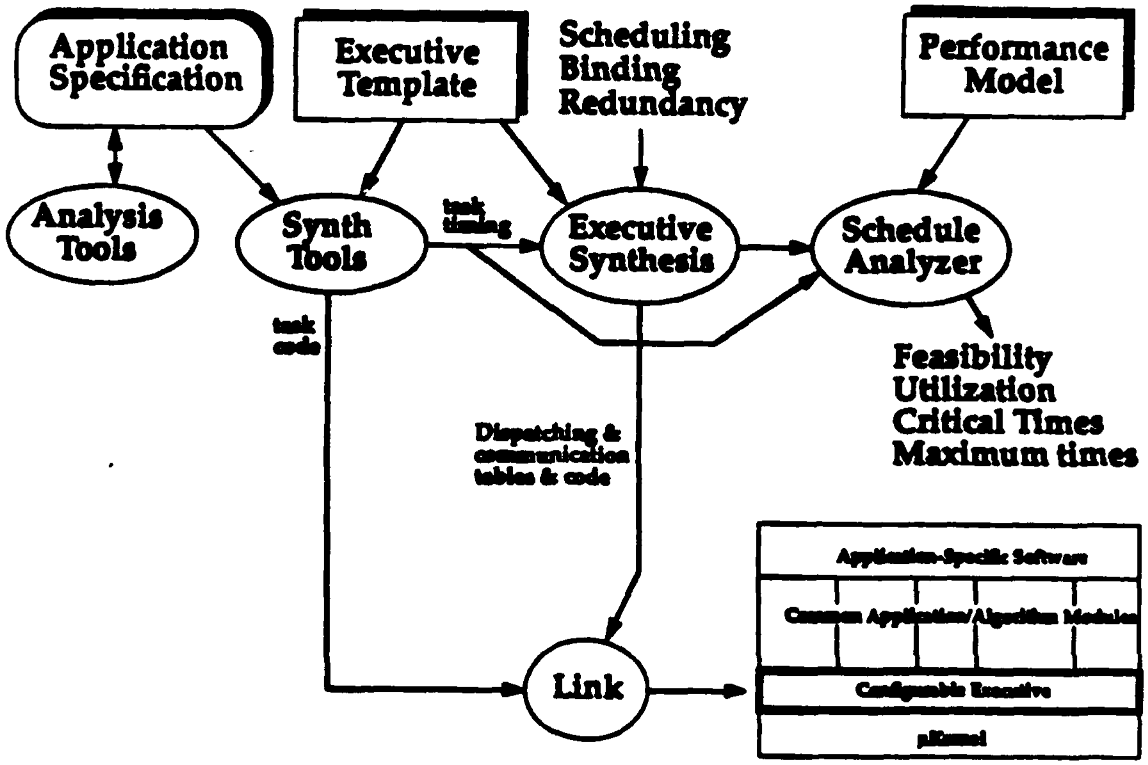

Application Specification
Executive Template
Scheduling Binding Redundancy
Performance Model
Analysis Tools
Synth Tools
task timing
Executive Synthesis
Schedule Analyzer
task code
Feasibility Utilization Critical Times Maximum times
Dispatching & communication tables & code
Link
Application-Specific Software
Common Application/Algorithm Modules
Configurable Executive
μKernel

Figure 2.4.1-2. DSSA Tools

2.4.2 V and V/Test

An evaluation of testing methods and tools was performed by Honeywell Sensors and Systems Development Center (SSDC) for SEMATECH in 1991-92. The study defined and classified testing tools specific to the SEMATECH software environment (semiconductor fabrication). Two specification testing tools were evaluated: SofTest and "T," both aimed at driving improvements in the specification and generating test cases for application to the software product.

Other V&V Test tools are available that are directed at structural testing of source code modules. Debuggers also fit in this category, as they are used in many organizations to verify correct execution of statements within a module.

2.4.3 *Programming Languages*

General-purpose programming languages (e.g., Ada, C++, Lisp) have been used extensively in the programming of signal processing systems. New languages such as Adapt and ASSIGN are emerging that incorporate domain-specific features that relieve programmers of making non-value-added (but error-prone) decisions during the coding process.

2.4.4 *Embedded Operating Systems*

Several operating systems targeted at embedded, real-time systems are commercially available at present. Many of these are either at or converging towards the POSIX interface standard (e.g., LynxOS, Real/IX, RTU, Chorus/MiX, OS-9 [RTUnix]. Few of these operating systems are geared toward distributed systems, however. Mach, an operating system substrate, is beginning to address this need by providing primitives that perform communications and signaling over a variety of distributed hardware architectures. Efforts are currently underway, funded largely by the Center for High Performance Computing (CHPC), to extend Mach's capabilities in the areas of transparent distribution, process migration, real-time scheduling, fault tolerance and security.

Another effort relating to the operating system environment is ISIS, developed at Cornell University. ISIS is a system for building applications consisting of cooperating, distributed processes. ISIS is currently being ported to the Chrorus and Mach microkernels.

2.4.5 *Micro-code Development*

The need for micro-code development depends on the existence of application-specific processors in the computing architecture. This need is diminishing as the performance of general-purpose processors continues to rise. Nevertheless, micro-code development support will be needed for those RASSP applications that require the use of special processors.

The current state of the art is represented in IDAS, mentioned in Section 1.1 IDAS provides Ada-to-microcode compilation for an arbitrary horizontal processor, including machine dependent optimization and compaction (achieving automatic-to-manual compactness ratios on the order of 1.5:1). IDAS will automatically create a simulator for a VHDL-described machine and will also automatically retarget the Ada-to-microcode compiler system to that machine; it will also generate a corresponding test and evaluation environment.

2.4.6 Software Technology Progression

In Section 3.4, Figure 3.4-1 shows a rough progression of technology in the area of software design. The first row addresses operating system technology, the second row microcode support, and the third row application development (this assumes a domain-specific software architecture is developed for RASSP).

Version 3.0 of Mach is scheduled to be released early in 1993. It will include features for "soft real-time," that is, coping with hard deadlines, but with relatively long context-switch times. Also scheduled for 1993 are soft, real-time inter-process communication (IPC) and instrumentation. In the 1994 time frame, the instrumentation features will become distributed and dynamic (active or "smart"). In 1994-95, Mach will incorporate hard real-time extensions, adhere to the Posix.5 interface standard, and acquire a security trust level of B1. In the out-years, fault tolerance will be incorporated into Mach.

Microcode compilation technology will most likely progress at modest levels through the 1990's, approaching hand-coded compaction levels for domain-specific architectures by 1996. The compilers will also be completely automatic retargeted based on VHDL descriptions of the hardware.

On the application development side, more and more of the software will be reused or synthesized from high-order design languages (HDLs). A critical enabler for this will be tightly integrated trade off analysis tools to assist in the system partitioning process DSSA.

2.5 Information Infrastructure and Data Base

In early 1992, The F-22 SPO began an effort to define a roadmap for an F-22 Integrated Weapon System Data Base (IWSDB). The purpose of the IWSDB is to enhance the accessibility of weapon system data, including engineering, manufacturing, logistics and management data, resulting in lower life-cycle costs and faster turnaround (see Figure 2.5-1). Configuration management is also a high priority.

The first of two workshops was held on May 12-14, 1992 to define the IWSDB's functional requirements, and a set of detailed process scenarios were developed. The second workshop, held on June 1-3, 1992 laid a framework for assessing architectural alternatives. The three fundamental alternatives were characterized as "integrated, federated, and mediated."

A fully integrated IWSDB would require all subsystems to agree on data access protocols and data structure definitions. Efforts along these lines are embodied by PDES (Product Data Exchange for STEP), an industry consortium for developing product data representation standards, and IDS (Integrated Design System), an Air Force program to develop a common backplane and process control system for design tools.

The F22 IWSDB effort is scheduled to establish a requirements specification, a reference model and a 10-year implementation plan by December of 1992, with implementation tasks to begin in early 1993. The Joint CALS office is actively coordinating with the F22 IWSDB effort, and actively participated in the second workshop.

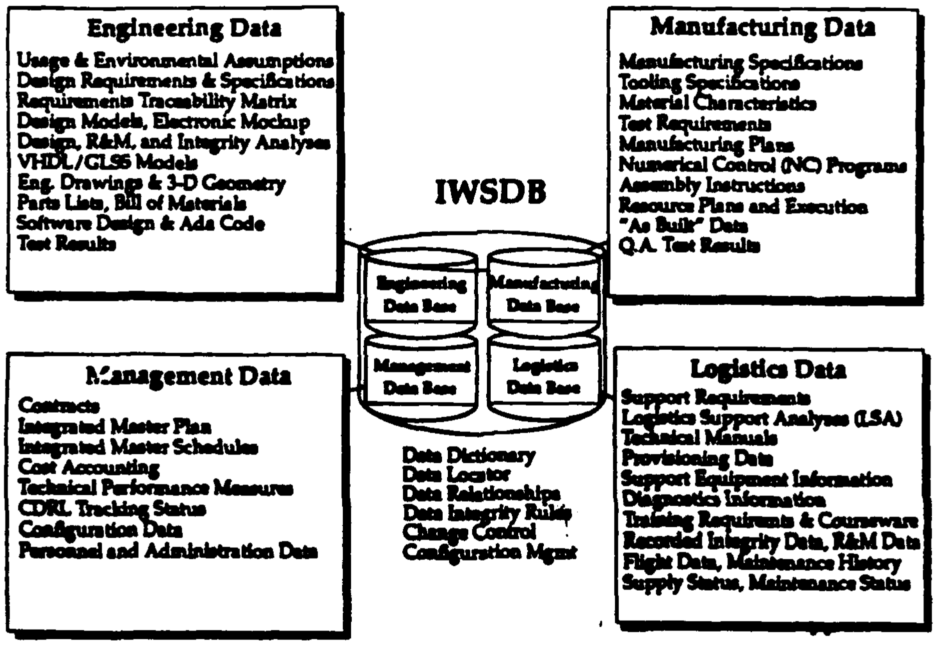

Engineering Data
Usage & Environmental Assumptions
Design Requirements & Specifications
Requirements Traceability Matrix
Design Models, Electronic Mockup
Design, R&M, and Integrity Analyses
VHDL/CLS6 Models
Eng. Drawings & 3-D Geometry
Parts Lists, Bill of Materials
Software Design & Ada Code
Test Results
IWSDB
Manufacturing Data
Manufacturing Specifications
Tooling Specifications
Material Characteristics
Test Requirements
Manufacturing Plans
Numerical Control (NC) Programs
Assembly Instructions
Resource Plans and Execution
"As Built" Data
Q.A. Test Results
Engineering Data Base
Manufacturing Data Base
Management Data Base
Logistics Data Base
Management Data
Contracts
Integrated Master Plan
Integrated Master Schedules
Cost Accounting
Technical Performance Measures
CDRL Tracking Status
Configuration Data
Personnel and Administration Data
Data Dictionary
Data Locator
Data Relationships
Data Integrity Rules
Change Control
Configuration Mgmt
Logistics Data
Support Requirements
Logistics Support Analyses (LSA)
Technical Manuals
Provisioning Data
Support Equipment Information
Diagnostics Information
Training Requirements & Courseware
Recorded Integrity Data, R&M Data
Flight Data, Maintenance History
Supply Status, Maintenance Status

Figure 2.5-1. F-22 Integrated Weapon System Data Base

There are a few types of data items relating to RASSP that apparently have not been discussed in the F22-IWSDB effort thus far, including:

- Interoperable analysis models (integrating hardware and software concerns)–These are clearly necessary for the rapid turnaround of signal processing designs, especially when upgrading fielded systems.

- Design rationale–This information is often inaccessible once a system goes into the field, which increases the potential for mistakes to be repeated.

- Engineering and manufacturing process models–Process models allow the explicit definition of *how* something is done. The F22 IWSDB appears to be focused primarily on recording the outcomes. Being able to "replay" a development process is an important consideration for RASSP.

- Linkages from advanced technology development (research) to engineering and manufacturing activities.

- Commercial business plans–For RASSP to be viable, it must be interleaved with commercial business processes. Linking business planning data with engineering and manufacturing analyses will play an important role in formulating bids on military programs.

A "data base" or repository to capture and track data is but one facet of the overall problem, however. RASSP must be concerned with the overall *infrastructure* which includes a variety of services in addition to data representation and access. Together, the list of infrastructure concerns is:

- Data services
- Process services
- Cross-functional tools
- Configuration management, traceability and change notification
- Reuse.

The infrastructure concerns are described in Table 2.5-1 with respect to state-of-the-art programs, standards, issues, and RASSP recommendations. These concerns are discussed in details in following subsections.

Table 2.5-1. Information Infrastructure Technology Issues

Technology	SOA	Pgms	Stds	Issues	RASSP Recommendations
Data Services	Relational, multi-base, namespace servers, object-oriented DBs, translators	Carnot, Extract, OOODB, POB, DICE, EIS	SQL(2), OOSQL, CFI, OMG, PCTE, IRDS	Distributed systems still in infancy; lacking O-O standards; poor performance over WANs	Track technology and product developments
Data Models	IDEF, Express, OOA, Application Protocols	PAP-E, ASEM, DICE	CALS, PDES, VHDL, IEEE-1175, CFI	Standardization requires time, investment	Develop SP application protocol; Support CFI, PDES
Process Services	state-based, coarse process enactment	EIP, DICE, Arcadia, STARS, EIS	CFI, OMG, PCTE	Parallel, domain-specific standardization efforts	Track technology and product developments.
Cross-functional Tools	Domain-generic CSCW; Reqs mgmt	DICE		Desigion rationale not captured	Stimulate CE-for-SP technology and product development
Reuse	domain-specific, keyworded code modules	STARS, ProtoTech, EIS-AD, CAMP	none	automation supports only partial reuse of configurations, little linkage to reqs, system design	Aim domain model towards reuse repository; establish RASSP library
Traceability, CM, Change notificationt	File-based CM, islands of fine-grained mgmt	ASID/VS, E-SLCSE		Discontinuity between disciplines; coarse change notification	Track F-22/ IWSDB development

2.5.1 Data Services

Data services can be further broken down into data description, and data storage and retrieval.

Data description is currently supported by tools such as IDEF, Express, and a host of Entity-Relationship tools. Many of these are integrated with data base management systems, giving the user the ability to automatically generate database schema code directly from the data description language. The challenge comes in trying to standardize such models.

PAP-E (PDES Application Protocol-Electrical), ASEM (Application Specific Electronic Modules), and DICE (DARPA Initiative in Concurrent Engineering) are examples of large government programs involved in standardizing various aspects of product definition data and associated access protocols. CALS, PDES, IEEE, ANSI and CFI (CAD Framework Initiative) are all focused on various elements of this standardization. As with PAP-E and ASEM, RASSP will likewise define requirements for communication information between process steps, and should invest in the standardization of data models that meet these needs. PDES and CFI are likely to offer the most payoff for such an investment.

Data storage and retrieval services are well on their way to industry-wide standardization. Languages such as SQL and SQL-2, based on relational storage technology, are already in widespread use and supported by an adequate number of vendors. Object-oriented database (OODB) products have been emerging during the last five to seven years and are finding niches in relationship-intensive disciplines such as engineering and manufacturing.

As one would expect, both standards and tool support for OODBs lags behind relational technology, but several programs are underway to improve both fronts. Microelectronics and Computer Consortium's (MCC) Carnot and Extract programs are pushing forward the functionality of OODB systems. DARPA's Open Object-Oriented Database System (Open-OODB) project is accelerating the standardization process for system interfaces and query languages for OODBs. ANSI has begun consideration of further enhancements to SQL to make it more suitable for use with OODBs. The Portable Common Tool Environment (PCTE), already a European standard, is growing in popularity as a platform for integrating design tools from a variety of disciplines including electronics, electronics manufacturing, and software.

2.5.2 Process Services

Packages like DMCS (Data Management and Control System) from Structural Design Research Corp. typify the state-based approach to process management, wherein the engineering process is driven by the "states" of various documents. A somewhat more flexible, event-oriented approach is evident in TeamNet, from TeamOne, Inc.

Virtually no standardization of framework interfaces for process management services has occurred to date. Effort is currently underway at CFI and OMG, and some candidate interface specifications have emerged, but they are still several months away from ratifying anything. TeamOne is opening the interfaces to their product beginning 1Q93 with a product release call the Process Automation Kit, or PAK.

2.5.3 Cross-Functional Tools

Two topics considered here are Computer-Supported Cooperative Work, and requirements management. The DARPA Initiative in Concurrent Engineering (DICE) is examining various approaches and tools for supporting "virtual tiger teams," multidisciplinary, geographically dispersed work groups. Some of the projects included under the DICE umbrella are the Design Notebook design, MONET , and a CE Blackboard.

Requirements management support is typically supported by a separate toolset and not well integrated with the design or manufacturing environment. New tools are emerging that link requirement expressions with design tools to dynamically evaluate the status of requirements refinement by the various design disciplines (e.g., CimflexTeknowledge)

A facet of the infrastructure that has not received much attention to date is in the area of decision rationale capture. Current research is focussed primarily on design decisions, but many other kinds of decisions carry a comparable or greater cost, schedule and/or quality impact, and the rationale for such decisions should be recorded for future reference. This "domain genericity" makes it a good candidate for inclusion in the set of infrastructure services.

2.5.4 Traceability, Configuration Management, and Change Notification

Traceability, configuration management and change notification have one thing in common; they all deal with interdependency relationships between data. Most commercial systems that support one or more of these facilities do so at the file level, that is, they maintain dependency relationships between whole files without attempting to resolve individual objects within a file. Examples of such systems are Sherpa's Design Management System (DMS), SDRC's Design Management and Control System, Digital's PowerFrame, and Hewlett Packard's SoftBench.

Domain-specific systems are beginning to break through the file granularity barrier, e.g., the Ada Software Integrated Development/Validation System (ASID/VS; Air Force, Wright Labs), the Knowledge-Based Software Assistant (KBSA, Air Force, Rome Labs) KBSA, and the Extended Software Life Cycle Support Environment (E-SLCSE, Rome Labs). These programs employ fine-grained repositories for storing, accessing and interconnecting data elements. An upcoming version of TeamNet (TeamOne, Inc.) will have a limited ability to resolve data items within a file.

2.5.5 Reuse

Not normally considered a part of infrastructure services, reuse technology has been gaining momentum as a set of techniques and methodologies to be applied to all aspects of the product definition process, from requirements on down to tests and logistics support.

Successful reuse methodologies to date are domain-specific, employing key-worded elements to facilitate retrieval in a given reuse context (e.g., CAMP). Newer reuse methodologies are based on a domain model [Holibaugh]. The infrastructure plays an important role in reuse, since it is the primary agent for widespread data access and contains the domain model. Engineering processes, controlled to some extent by the infrastructure, can be constrained to proceed with a bend toward reuse EIS-AD.

2.5.6 Framework Evaluation

Several products are currently available that can serve as infrastructure backbones, or frameworks. Each one operates best in a particular domain, e.g., software development, electronic design, etc. (See Table 2.5-1)

Table 2.5.6-1. Evaluation of Existing Frameworks

Product:	Granu-larity	Specialization	Maturity	Data Model	Process Model	Standards
Teamnet/ Teamone	File	Change management	Med	Files only	Limited	
DMCS/SDRC	File	Shell for eng data	High	Files only	Rules	
DMC/Sherpa	File	Engineering data/ Change Mngmnt	High	Files only	Programming Language inter-face	
Softbench/HP	File	Software control	Med	Files only		
Falcon/Mentor	Object	ECAD	Low	Special License	Integ new tools/ Define new con-trol flows	Tracking CFI
Cohesion/Digital	Object	Software design	Low	Extendable	Special interface	ATIS based[a]
Backplane/ Atherton	Object	Software design	Med	Limited new object classes	Integ new tools/ Define new con-trol flows	ATIS based
PCTE - EAST - E2	Object	Business software	Low (USA)			European, vari-ous US efforts

a. moving toward PCTE

2.6 Test and Evaluation

In the design and fabrication of RASSP, the task of test and evaluation is of paramount importance. The interactions of the testing task with all other activities during the RASSP development is presented in Figure 2.6-1. If not well planned out and designed ahead of time, the time and cost taken for fabrication and testing of the design and product can be very significant and can well negate most of the advantage gained by the use of RASSP methodology.

The requirement imposed on the testing task is determined by the RASSP product characteristics. The product system to be developed in RASSP is envisioned to be made up of advanced ICs with high chip density and high speed clock rate. New packaging technologies such as multi-chip module (MCM) and high-density interconnection (HDI) are expected to be used. This means that the component and assembly costs increase substantially with each level of integration. Economy dictates that reworks due to defects should be carried out as soon as they are detected and as early on in the integration as possible. This implies extensive testing and evaluation are required at each level to assess the integrity of the components/product and expose any defects that may exist. On the other hand, in order to support the implementation of the "model year" concept that is central to the RASSP program, the cost and time of the test effort must be minimized. This includes not just the amount of actual testing time, but also the cost and effort involved in the test generation itself. The RASSP program thus calls for extensive testing but with minimum time and cost.

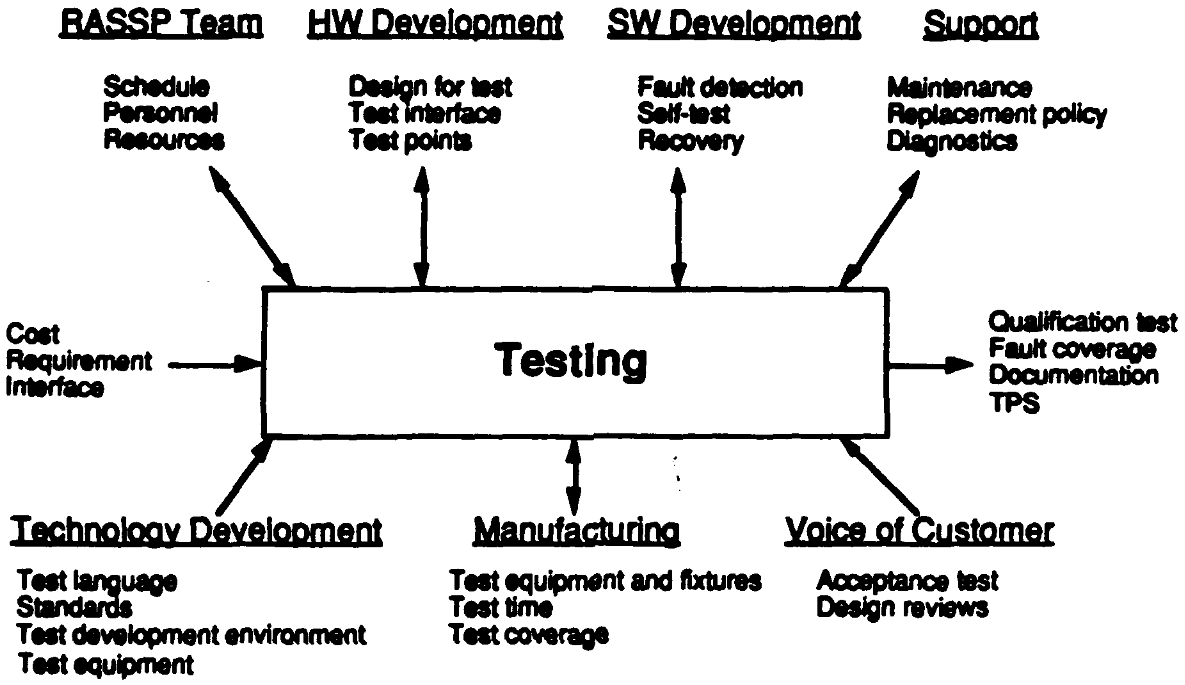


Figure 2.6-1. Input and Output for Testing Task

In order to achieve the above objectives, a number of test approaches are recommended for use in the RASSP methodology. When taken together, they offer the best chance in ensuring the successful implementation of the test and evaluation task. These approaches are listed here and discussed briefly in the following subsections :

- Hierarchical testing;
- Design-for-testability;
- Testability analysis;
- Performance monitoring and instrumentation;
- Integrated test support environment;
- Test data for design improvement.

2.6.1 Hierarchical Testing

Test problems can easily delay the introduction of a new product and thus affect the realization of rapid prototyping and "model year" concept. Testability and diagnosis must be planned early in the design cycle; otherwise it may prove to be very costly, or even impossible to accommodate later. Test cannot be treated as an afterthought in the design process. A properly designed test strategy will also facilitate field diagnostics for the system during its life cycle.

A system produced in the RASSP program is assumed to consist of a number of boards, each of which is made up of several MCMs. The MCM in turn is made up of a number of bare ICs. A hierarchical testing strategy is therefore the most logical approach to take. Experience in manufacturing industry has established a general rule of thumb that states that the test and repair cost goes up by an order of magnitude as testing goes up by one level. Suppose a defect exists in a bare IC, if it can be tested, detected, and repaired (replaced) at the lowest level (IC level), and a certain cost, say x, can be associated with it. If the same defect is not detected until the next level up (MCM level), then the test and repair cost will escalate to 10x. This cost will increase again similarly as the test goes up to the board level and so on, finally arriving at the system level. Therefore, it makes economical sense to perform extensive testing at the lower levels to detect as many defects as early as possible.

Figure 2.6.1-1 illustrates the recommended testing hierarchy for a RASSP system. At the IC level, extensive testing including structural, functional, parametric, electrical, and temperature tests are performed. The known acceptable ICs are then mounted together to the package substrate to form MCMs. Each is then tested as an individual unit. At this level, new tests such as new functional test, scan path test, and interconnect test, will also be included. Even before the costly ICs are mounted, the substrates need to be verified for electrical integrity and probed for shorts and opens. The MCMs that successfully pass the screening test are assembled and mounted onto the printed circuit boards (PCBs). Again the PCB connectivity is tested prior to the components being mounted. At the board level, only the functional test and board acceptance test need to be performed. When the boards are assembled with the necessary backplane and chassis, they will form a system. Tests at this level generally include backplane test, functional test, and qualification test with a system testbench.

As shown in Figure 2.6.1-1, the test results and data obtained at each level are passed up and used to support the testing task at the next higher level in a coordinated manner so that components need not be retested unnecessarily each time. Another important consideration for implementing a hierarchical testing approach is that requirements imposed by higher level tests must be made known and specified during the design of the components at the lower levels. For example, if boundary scan is planned to be used as part of self-test for a specific MCM, then the individual IC design must include the necessary control interface circuit even though it may not be needed for testing at the IC level. This means that appropriate design-for-test features are incorporated into the design at each level to support a comprehensive overall hierarchical testing strategy for the system.

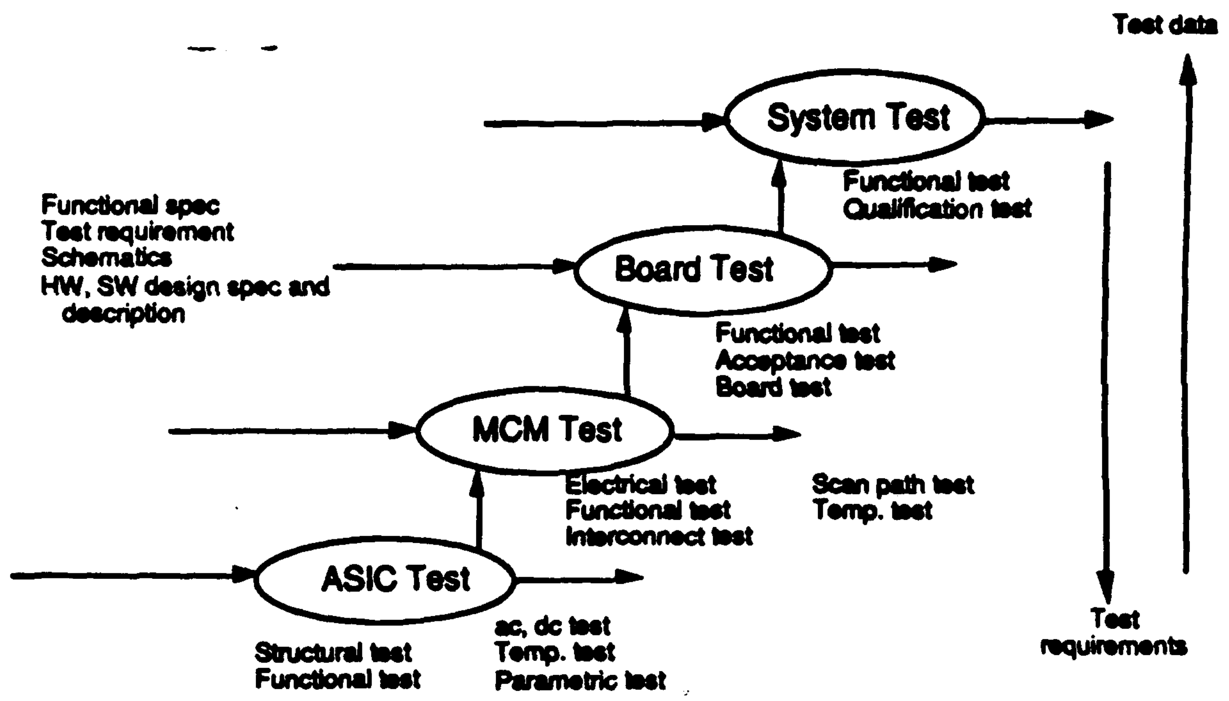


Figure 2.6.1-1. Testing Hierarchy

2.6.2 Design-For-Testability (DFT)

Conventional testing approaches are no longer adequate for the RASSP program. The RASSP program will be using the new emerging packaging and interconnect technologies such as MCM, HDI, or even the exotic three-dimensional methods, in addition to the classical printed circuit boards and backplanes. Because of the high chip density and the small interconnect line dimensions (line widths in the range of 15 to 50 microns, about an order of magnitude smaller than that on PCB), the traditional bed-of-nail testers cannot be easily used to gain access to internal lines. Even if that becomes possible, this type of tester will be extremely costly. In addition, external testers are rapidly approaching another physical limitation, the circuits under test may be faster than the delay along the tester probe lines. While packaged devices can be tested at over 100 MHz, it is nearly impossible to adequately test bare die to obtain known acceptable ICs prior to their assembly into an MCM. This necessitates the incorporation of in-circuit testing through use of DFT and Built-In Self-Test (BIST) methods.

As the complexity of the system increases, generating tests for them has emerged as a major problem and cost concern. The use of DFT is a very efficient way to convert larger complicated sequential circuits into a number of smaller combinatorial circuits. This in turn allows the use of automatic test pattern generation. The number of test vectors required to achieve a specified fault coverage will also be greatly reduced.

The objective behind the use of DFT is to achieve controllability and observability on the internal states of a system to make it easier to test the system. Large complex sequential circuits are typically partitioned to form smaller combinatorial circuits by adding extra control logics to storage elements (such as flip-flops) in the circuit and connecting them together to form serial shift registers or scan paths. Each of these storage elements can then act as a pseudo input or output port. Variations in the interconnect patterns and controls of these storage elements can be used to form different pseudo-random test vector generators and signature analyzers for use in built-in logic-block observation (BILBO) and self-test.

Currently there is not a standardized way of implementing the scan paths and control logic, although IBM's Level Sensitive Scan Design (LSSD) method remains the best known approach. Efforts to standardize DFT approaches have been attempted by DoD, resulting in the development of PI Bus Test and Maintenance Bus. These efforts in turn have led to the formation of the IEEE Standard 1149.1 on boundary scan in 1989. This standard has now been widely accepted and used in the CAD/electronics design industry. Other 1149.x standards are currently being defined and worked on by the design and test community to deal with analog signals, mixed signals and other test interfaces.

A wide array of DFT techniques and features can be chosen for use in designing testable systems, each of them has its own unique capability, cost and ramification when used in a particular design. A designer will have to select intelligently an optimum set of DFT techniques to form a comprehensive test strategy for meeting his testability requirements. Testability must be included at each level of design for the entire system to be tested efficiently, from the system level down to the bare die level. There must be a consistent design for testability philosophy throughout the different levels so that features and techniques used at each level can be used to cooperate and complement each other in a cost effective manner.

2.6.3 Testability Analysis

The incorporation of testability in the design phase ensures that a system can have the inherent ability to be testable, so that desired levels of fault detection and isolation can be achieved in an accurate, timely, and cost-effective manner, and thus reduce its life cycle cost. The task of evaluating and assessing the appropriate testability requirements in a design is greatly facilitated by the use of computer-aided design (CAD) tools for testability analysis. This is especially appropriate in the early stages of the design phase when many options are available and designs can change rapidly.

A testability analysis tool is used to evaluate the design information and provide a quick assessment of the inherent testability of the system, as well as insight into the details of the testability in specific portion of the system or components. The output information should include quantitative results such as the various ambiguity groups, feedback loops, types of testing strategy to apply, and qualitative results such as suggestions to improve the design. These may include recommendations on where to break feedback loops, where to add test points, or where to eliminate test points.

The importance of performing testability analysis during the design phase of a system is underscored by the emphasis that DoD has placed in requiring this analysis be performed in Mil Std 2165 (Testability Program for Electronic Systems and Equipments) and Mil Std 1814 (Integrated Diagnostics).

A number of CAD tools on testability analysis are currently available on the market. These include:

- CAFI—Computer-Aided Fault Isolation and Testability Model from ATAC of Mountain View, CA.
- STAT—System Testability Analysis Tool from Detex Systems, Inc of Orange, CA.
- STAMP—System Testability and Maintenance Program from ARINC Research Corporation of Annapolis, MD.
- I-CAT—Intelligent Computer Aided Testing from Automated Technology System Corporation.
- WSTA—Weapon System Testability Analysis tool from Harris Corporation.

A study comparing the various tools mentioned above indicated that at the present time, STAT seems to be the most useful testability tool to be used in the design phase as it possesses all the essentials to aid in the design of testable systems. Its numerous user options allow a designer to make use of "What if" type scenario. The numerous test node suggestions and the interactive feedback loop breaker make STAT a suitable tool to aid in the design of testable systems. STAT also provides a figure of merit in terms of fault isolation and ambiguity groups.

Besides testability analysis tools, other CAD tools are beginning to emerge for helping designers to choose or incorporate DFT techniques. Commercially available tools include:

- Test Assistant Tool from VLSI Technology
- TEXPERT(Test Expert) from GTE Labs
- Test Compiler from Synopsys
- Intelligen from Racal-Redac/HHB
- ScanGen from AIDA/Teradyne
- Design Advisor Tool from NCR.

Most of these tools are either general design rule checkers that verify the conformance of a low-level structural design to prespecified ASIC design rules or DFT synthesis tools. The common limitation among all these tools is that they do not provide any guidance to help the designers to decide what type of DFT is needed and how much is needed. Also they typically all follow very rigid structural rules without paying too much attention to the design functionality or how the DFT structures may be shared among different portions of the design. For example, all the full-scan insertion tools generally adhere to very simple transformation rules and do not provide a mechanism for making trade-offs or to accommodate design constraints. This is too inflexible in most cases, and may result in overly cost design.

Other more flexible and intelligent tools are currently being developed in the Universities and research organizations. These efforts seems to be addressing many of the needs that designers have in generating more testable systems. Some of these systems may eventually find their way into the commercial world yet. These include:

- TDES (Testability Design Expert System)—University of Southern California
- TIGER (Testability Insertion Guidance Expert System)—MCC
- DEFT (Design for Testability Expert System)—Purdue University
- Frenchip—Dassault Electronique of France.

2.6.4 Performance Monitoring and Instrumentation

Many of the target systems (such as the Touchstone System) being considered for the RASSP program will utilize parallel and distributed processing architectures and large amounts of software to accomplish demanding application functions. Just testing the individual hardware and software component modules by themselves will not be sufficient. The entire system, with its hardware, system and application software, must be tested and verified as they work together in real-time operation. As the architectures and algorithms become increasingly complex, it becomes extremely hard to accurately determine how a particular piece of hardware and software design performs in the presence of all the other components in the system.

A tool for instrumenting and monitoring these parallel and distributed system is needed to implement stimulation functions such as synthetic load and fault-injections, as well as monitoring functions such as detecting performance bottlenecks, quantifying real-time performance, detecting and isolating behavior anomalies. Performance monitoring, debugging, and evaluation techniques for uniprocessor systems have been in practice for a long time. Techniques ranging from hardware-based memory-bus monitoring to instruction tracing are typically used to provide program execution profiles, timing and traces. With parallel systems, we now will have to add high-level information about interactions among distributed hardware and software entities.

Recent attention has been paid to instrumentation for parallel systems with emphasis on performance measurement and evaluation of software entities, and on built-in instrumentation at the operating system level. Techniques used include:

- Interactive performance monitoring/analysis;
- Animated real-time displays of distributed system state and interactions;
- Advanced parallel debugging and performance analysis.

Honeywell is currently developing a comprehensive testbed instrumentation approach based upon the concept of events, actions and event-action virtual machines. In addition to statistical performance monitoring, the following instrumentation functions can all be uniformly implemented as actions executed in response to events:

- Behavioral measurement;
- Data reduction;
- Experiment control;
- Stimulation;
- Experimenter interface.

The event-action machines are typically implemented in software at the operating system and application levels, but can also be implemented partly in programmable hardware to achieve better response time and throughput for certain critical instrumentation functions. Current phase provides monitoring capabilities in a minimally-intrusive manner. Future phases will add non-intrusive monitoring, stimulation and control capabilities.

2.6.5 Integrated Test Support Environment

In order to successfully implement the "model year" concept that is central to the RASSP program objective, the traditional approach used in the test development process will have to be replaced by a new approach in which walls between different engineering and test groups are removed so that they can work together in a concurrent manner. The traditional approach has been basically an "over-the-wall" approach in which one group of engineers would work more or less independently on one aspect of the test development process, and then pass its work and results on to the next group for. Thus the design, test requirements definition, test program set development, and automatic test equipment definition tasks are all being performed as distinct and separate stages in the test development process. Changes made in any one of these stages typically would require much paper work and time to ripple through to the other stages, and for feedback to return. Honeywell is currently developing a new approach that will enable these functions to be performed in an integrated test support environment, by removing the "walls" between them through information sharing.

The objective of this new approach is to provide a significant improvement in cost, schedule and quality of the test process by integrating product engineering, test engineering, logistics and manufacturing functions concerned with test. The primary focus of this approach is to institute a process and environment to support concurrent development and cross-functional sharing reuse of test requirements and related information between the above mentioned groups, coupled with an integrated change control process. Changes to the product are typically made based on the desire for additional performance or to correct problems without considering the impact on the support infrastructure. The integrated test support environment will provide the capability to concurrently assess and deal with impacts as changes are proposed to product requirements. Identifying, understanding, and planning for change are basic principles that must be utilized.

The test development process typically involves the following functions:

- Identify product requirements
- Define test requirements
- Develop test programs
- Collect data
- Identify problems
- Analyze problems
- Incorporate changes
- Track changes

To support the above functions, the integrated test support environment will have to rely on infrastructure, models, tools and a well-defined process.

The infrastructure provides the basic resources upon which specific tools and processes are implemented. It must include:

- An integrated, high performance network file system;
- A federated set of information management systems integrated through a global data dictionary/cataloging system;
- Process management services; and
- Tool integration services.

Three forms of models will be required to define and implement the needed integration:

- Information model —to define all relevant information entities and artifacts, and critical relationships between them.
- Implementation model—to capture the implementation of the information entities defined with the information model. The information model and its implementation will form an information framework that will support the integration and evolution of additional information standards such as TRSL, ABET, PAP-E and AI-ESTATE.
- Process model—to define the engineering and change process in the selected domain. This model will focus on the life cycle of the artifacts of interest as well as the relationship between transitions between life cycle steps and activities, personnel, and mechanisms used to perform the transitions.

A set of integrated tools, based on commercially available tools, should at the minimum include:

- Requirements identification and tracking tools;

- Problem reporting and change tracking tools;
- Test requirement authoring tools;
- Test program generation tools.

The requirements and change process must be clearly defined to leverage the enhancements in infrastructure and tools to implement a more expeditious change process. Areas that must be addressed include:

- Early identification of problem impact capabilities during problem analysis.
- Increased concurrence in change incorporations due to early identification of change impacts. This is in contrasts to the current process where initial changes are identified and subsequent impacts are considered as changes are made, thus resulting in a difficult to control or predict cascade of changes.
- More cost and schedule conscious planning for change.
- Integration of the change process between prime and subcontractors.

2.6.6 Test Data for Design Improvement

Data from testing and production must be collected and utilized in the RASSP program to improve the design and reduce time and cost for the next "model year" product. Test data includes data results from the test development process as well as test execution.

During product design, test requirement envelopes will be defined and analyzed to uncover inconsistencies and gaps. Product dependencies and performance characteristics conveyed by product description will be analyzed to identify test problems.

Manufacturing tests will be instrumented as part of test program generation to collect test results. These results will be analyzed to identify problems, such as performance characteristic that is consistently different than nominal. Traceability from product performance descriptions through test requirements and implemented tests will aid in associating problems with causes. These test results can also be analyzed to identify manufacturing process problems.

This concept can be extended to analyze test data collected during depot-level and organizational-level testing. This will allow identification of performance degradation problems that may be caused by design problems, such as thermal problems.

2.6.7 Testing Technology and Issues

This section discusses the major testing technology areas that are of relevance to the RASSP program, together with the issues and risks that need to be addressed, and some possible actions to mitigate these concerns. We have categorized the testing technology into three main areas;

design, test equipment, and test development. The current state of the arts in each of these areas, and the on-going or potential government programs and standardization efforts on them are also provided where possible. A list of the acronyms used for these programs are given in Table 2.6.7-1.

Table 2.6.7-1. List of Acronyms

ABET	Ada Based Environment for Test
AI-ESTATE	Artificial Intelligent Expert System to ATE
ASEM	Application Specific Electronic Module
CASS	Consolidated and Automated Support System
HITS	Hierarchical Integrated Test Simulator
ITSE	Integrated Test System Environment
IFTE	Integrated Field Test Equipment
LSSD	Level Sensitive Scan Design
SCPI	Standard Commands for Programmable Instrument
SS/REC	Simulatable Specification for Rapid Electronic Change
STAT	System Testability Analysis Tool
TDL	Test Description Language
TISSS	Tester Independent Support Software System
TRSL	Test Requirement Specification Language
VTEST	Virtual Test
WAVES	Waveform and Vector Exchange Standard
WSTA	Weapon System Testability Analysis

2.6.7.1 Design—As mentioned in the previous sections, testability must not be treated as an afterthought when designing a system. It must be designed in right from the start. This means performing testability analysis as part of the system design, incorporate design-for-test techniques in the design process, and making use of automated CAD tools such as test synthesis and test pattern generators as much as possible.

For performing testability analysis at the system level, a number of CAD tools are available commercially, such as STAT and WSTA. Both of them are based on dependency modeling. At the chip level, most of the tools are still at the research level and exist at the universities and laboratories. These include TDES (USC), DEFT (Purdue) and TIGER (MCC). DoD has established Mil Std 2165 to ensure contractors perform system testability analysis as part of the requirements they must meet. The design/test community is currently working on standardizing a data format for performing testability analysis through the AI-ESTATE program. The major issue in this area is that all these current tools only make use of topology and structure information,and are, therefore, limited in their capability in performing design trade-offs. For more comprehensive testability analysis, multiple-domain information models must be developed for use. DARPA programs such as ASEM may be the vehicle to provide the needed resource to develop such a comprehensive testability analysis tool. At present, only dependency-based models such as STAT and WSTA are available. We expect that by 1994, standard functional dependant models will be developed through efforts by AI-ESTATE. Hopefully, by 1996, multi-domain (including that of the ATE target) testability analysis can be implemented.

A number of design-for-test techniques have been developed over the years, such as LSSD, built-in self-test and boundary scan. Thus far the only standard that has been approved for use is the IEEE 1149.1 standard for boundary scan. There is as yet no standards for analog and other DFT cases, although efforts to develop them are underway and additional future standards are expected.

As mentioned earlier, a number of test synthesis tools are now available from CAD design tool vendors, such as Test Assistant (VLSI), Texpert (GTE), Intelligen (Racal-Redal/HHB) and Test Compiler (Synopsis). They are being used to provide help and guidance to the designers for inserting DFT features into design. The main issue faced in this area is that most of these tools are quite inflexible and rigid. They do not provide users with the choice and selection to make a design more efficient and cost effective. Research is currently in progress in universities and industry to overcome these restrictions.

Another design area that has been greatly benefited by the use of CAD tools is in the test pattern generation and fault coverage grading. Tools from Cadence (Test Scan), Mentor and Synopsys are now available for use for designs that incorporate DFT. Although they are still very computational intensive, the problem can now be alleviated somewhat through the use of hardware accelerator or distributed processing with multiple processor systems.

2.6.7.2 Test Equipment—In the "model year" concept, in addition to planning for specific upgrades to the product each model year, the production and support capabilities must also be upgraded to deal with the upgrades. The manufacturing environment is characterized by high mix and low volume. Test setup/teardown times in this scenario become important. This in conjunction with the concept of a "model year" and the steady change this entails requires flexible and easily reconfigurable test equipment as well as automated development tools.

Test equipment that supports steady increase in capabilities under a common architecture needs to be developed and installed. A test development environment that supports domain-specific requirement modeling and reuse, couple with automatic program generation targeted to the flexible ATE is also required. This domain specific approach should be pushed as far towards the product as possible.

At the ASIC level, testing is performed using commercial tester, electron beam probe, conventional test probe and DFT techniques. The IEEE 1149.1 standard on boundary scan has become a useful feature in testing. The main issue facing test at this level is that speed, pin count, and chip density are ever increasing. This has placed tremendous requirements on the capability of the test equipment. This is further aggravated by the need of testing bare die to obtain known good ICs for MCM. The incorporation of DFT has been very useful in simplifying the device pre-test and assembled MCM diagnostic testing. Also the commercial vendors and semiconductor manufacturers are now cooperating to a much greater extent than before to share test information and results.

MCM level testing is still relatively immature. The same equipment as used for ASIC is typically applied. More use is made of scan capabilities and in- circuit test at this level. Testing includes electrical, functional, interconnect, and temperature tests. The issue remains that of speed and the lack standard test interface besides that of boundary scan.

Board level testing includes functional, interconnect, and acceptance testing. Bare boards are tested prior to assembly. Testing at the board level also takes advantage of boundary scan capabilities where available.

System level testing includes functional and qualification tests. Testers applied here include commercial as well as custom designed automatic test equipment (ATE). The main issue at this level of testing is the absence of a standard architecture for ATE that can lead to a proliferation of ATE architectures. Efforts are now underway to develop common ATE standards at various software and hardware levels through programs like CASS and IFTE.

2.6.7.3 Test Development—The test development effort encompasses the generation of test requirements and specifications, test simulation of good and bad circuits, and test program generation.

Test Engineering is responsible for developing tests used in manufacturing while Logistics is responsible for developing test requirements documents (TRDs) for field-level testing. Currently, Test Engineering and Logistics cannot adequately share test requirements information, resulting in redundant efforts. Sharing would allow Logistics to take advantage of the test maturation achieved in manufacturing and manufacturing to leverage additional diagnostic tests added by Logistics. Organized reuse of test requirements would also allow expertise to be focused on developing high quality test requirements that would lead to a reduction in test-associated scrap and rework. An integrated process that provides the ability to engineer product changes and test changes concurrently, coupled with the ability to rapidly deploy the defined changes is essential to decreasing product cycle time and associated costs. Change occurring in a concurrent process with information sharing requires that all impacted functions cooperate in the change. This in turn requires that the dependencies between objects from the originating requirements to the lowest level implementation be identified, understood, and assessed.

A number of efforts are underway to take advantage of standards in test requirements and specifications, such as ABET, TRSL, WAVES, Mil Std 1519x, and Mil Std 1345x. One of the limitations with current test requirements is the lack of formal traceability to the specification. There is no traceability from a change request to affected components, test documents and procedures. Ideally test requirements are automatically extracted from product specification. One approach proposed by Honeywell to eventually accomplish this is to develop an integrated test system environment, and with the use of test requirement models to support analysis. Potential programs that can address this issue include VTEST, SS/REC, and TISSS. At present, test requirements and specifications are typically produced manually with guidance from Mil Std 1519. We expect that the technology in this area will progress such that by 1994, requirements and change tracking from systems requirements through implementation code can be accomplished (on EIP program), and that formal models and language can be developed to achieve requirements level reuse. By 1996, we expect the capability to extract test requirements and specifications from product models (on TRSL and VTEST programs) and perform system level simulation of test requirements (SS/REC).

In the area of test simulation for good and bad circuits, a number of commercial simulators such as LASAR and Mentor are now available for digital systems. However, tools for analog and mixed signal simulations are still lacking. Standards that have been developed or in development for use in test simulation include WAVES, VHDL, MHDL and AHDL. Programs like SS/REC will address some of the issues in this technology area.

The issue of concern to the area of test program generation is the development of tester-independent specification and capability of rapid development through reuse. Both of these objectives will result in significant reduction in the efforts for test program generation, and thus savings in time and cost. Programs like VTEST and TISSS are addressing many of these issues. The recommended approach for RASSP is to formulate a domain specific modeling approach, develop a model library for reuse and to leverage the efforts from programs such as ABET, VTEST, and SS/REC. At present, there exists some automatic test program generation capability, most are done manually. Few standards exist, most notable being WAVES. By 1994, we expect that ABET will have defined an architecture for target support. Architecture-specific test requirements driven generation can be implemented for program reuse. By 1996, we can expect that multi-target automatic test program generation from requirements and product models can be realized (through VTEST and TRSL programs).

2.7 Manufacturing, Design/Manufacturing Interface, and Cost

Manufacturing as addressed in this study primarily has been concerned with printed circuit board assembly. We have assumed that IC processing and ASEM/MCM fabrication are peripheral to this study and to RASSP manufacturing in general. ASICs and ASEMs are assumed to appear to a RASSP factory and its integral, sustained manufacturing/assembly as vendor supplied parts.

Presently, the majority of manufacturers integrate software with hardware as part of testing. The software development is part of the engineering department. Therefore, the software design technology described in Subsections 2.4 and 4.1 overlap the manufacturing process. The software testing, reliability, quality and configuration control are part of the design process. Even the improvements and corrections recommended by the field support are implemented by the field support, are implemented by the same software engineering group. A good example of state-of-the-art software design/manufacturing is given by the CAD vendors.

2.7.1 RASSP Manufacturing Methodology

The process steps comprising the recommended methodology for RASSP manufacturing are shown in Figure 2.7.1-1. The critical differences between the implementation of this process for RASSP manufacturing compared with traditional manufacturing are that it is

- Highly automated, planned for high volume assembly, yet amenable to a lot size of one;
- Designed to accommodate change by employing agile, flexible manufacturing techniques;
- Implemented in an organization committed to concurrent engineering/integrated product development "best practices" and to ongoing process improvement.

Manufacturing must be represented on the RASSP cross-functional team and must participate in initial product planning. This will ensure that producibility considerations are part of the product planning process and that manufacturing is a full partner in the entire product development process.

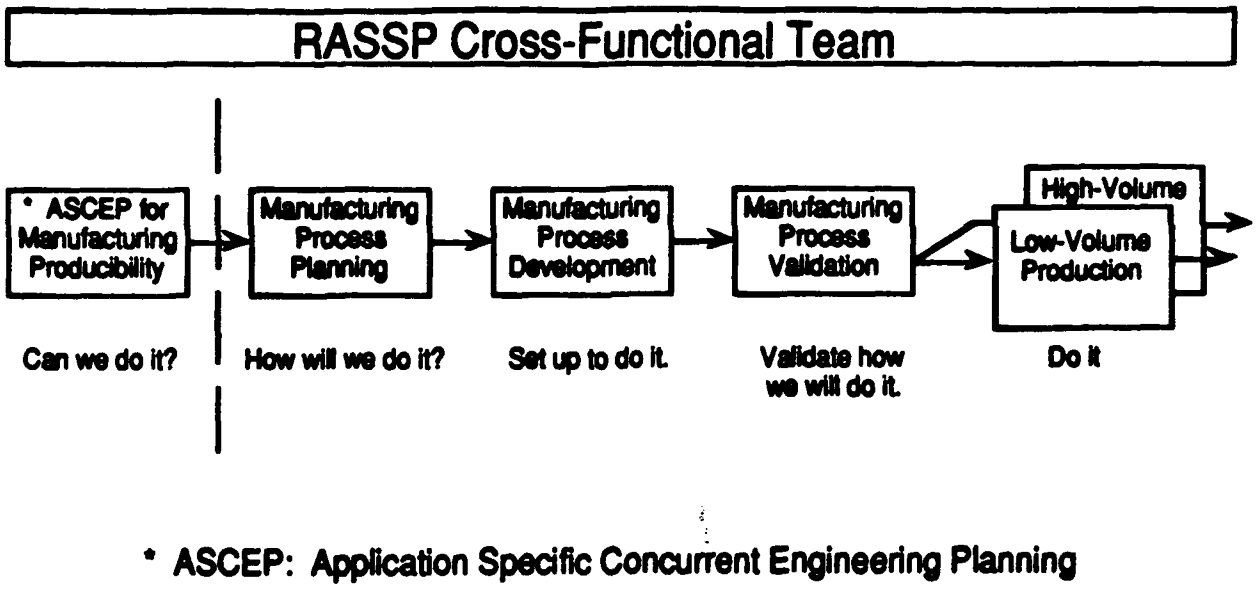

RASSP Cross-Functional Team
* ASCEP for Manufacturing Producibility
Manufacturing Process Planning
Manufacturing Process Development
Manufacturing Process Validation
High-Volume
Low-Volume Production
Can we do it?
How will we do it?
Set up to do it.
Validate how we will do it.
Do it
* ASCEP: Application Specific Concurrent Engineering Planning

Figure 2.7.1-1. RASSP Manufacturing Methodology

2.7.2 Design/Manufacturing Interface

2.7.2.1 Manufacturing-to-Design—Traditionally the design/manufacturing interface has emphasized the flow of information from design to manufacturing. However, as manufacturing technology has advanced, the flow of information from manufacturing to design has become equally if not more important. This is very evident in IC fabrication where process design rules must be established before circuit design can proceed. The same is becoming true at all levels of the system heirarchy as system performance and enabling technologies advance. Thus, at high clock rates and fine pitch metal spacings, PCB assembly must establish design rules and practices that constrain design.

Key considerations at this interface are shown in Figure 2.7.2.1-1. In addition to the design rules, the assembly line specification can identify a list of standard parts that will be available on the line. This will help reduce the cost of parts, especially if these standard parts are identified as group technology applying across an entire family of products. Rules for producibility can also be included in this specification.

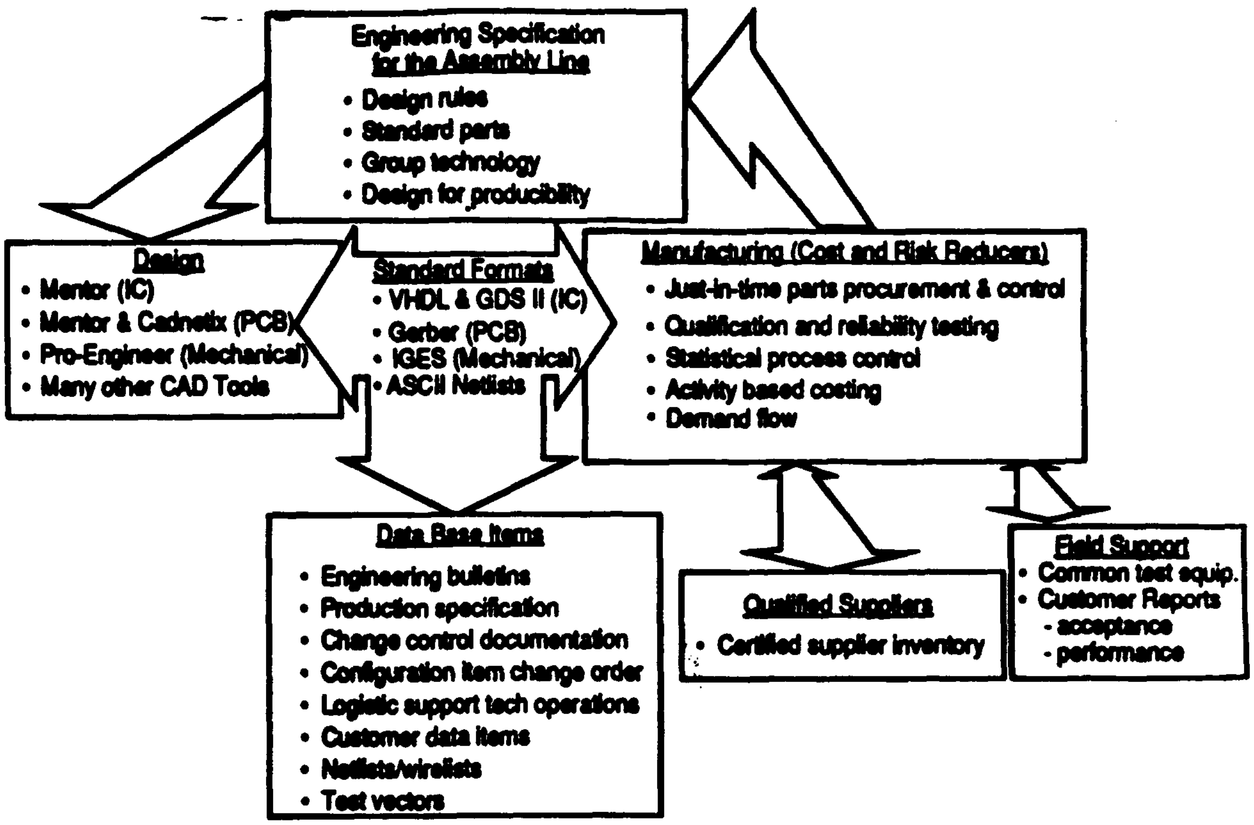

Engineering Specification for the Assembly Line
• Design rules
• Standard parts
• Group technology
• Design for producibility
Design
• Mentor (IC)
• Mentor & Cadnetix (PCB)
• Pro-Engineer (Mechanical)
• Many other CAD Tools
Standard Formats
• VHDL & GDS II (IC)
• Gerber (PCB)
• IGES (Mechanical)
• ASCII Netlists
Manufacturing (Cost and Risk Reducers)
• Just-in-time parts procurement & control
• Qualification and reliability testing
• Statistical process control
• Activity based costing
• Demand flow
Data Base Items
• Engineering bulletins
• Production specification
• Change control documentation
• Configuration item change order
• Logistic support tech operations
• Customer data items
• Netlists/wirelists
• Test vectors
Qualified Suppliers
• Certified supplier inventory
Field Support
• Common test equip.
• Customer Reports
- acceptance
- performance

Figure 2.7.2.1-1. RASSP Design/Manufacturing Interface

2.7.2.2 Design-to-Manufacturing—The flow of information from design to manufacturing for PCB assembly is basically electrical, mechanical, and test related information. Although many different CAD tools are used, all are commercially available and hosted on multiple platforms, typically Sun, HP/Apollo, and PCs. At Honeywell, PCB design data is most often transfered in Gerber format for board electrical layout and IGES for mechanical drawings. Mentor's BoardStation is Honeywell's preferred tool for PCB design, this tool being endorsed by Honeywell's corporate CAE committee. Cadnetix hosted on a PC and Dazix/Intergraph PCB software are also used by some design groups within Honeywell. Board artwork in Gerber format is electronically transferred to board manufacturers via modem/telephone link by several Honeywell divisions. In one case where the volume is high enough to justify it, a PCB manufacturer maintains an office within the Honeywell division and performs design rule checks on the Gerber data locally at Honeywell before transferring the data via modem/telephone link to his home office manufacturing plant.

2.7.2.3 Design/Manufacturing-to-Data Base—

Computer networking/sharing of design data is commonplace between Honeywell divisions, and as the above example shows, it is also done with outside vendors. However, outside companies would at best have controlled, limited access to internal Honeywell data bases and clearly the protection of company proprietary data is a real concern with any computer network linkage.

The solution that has been implemented on a local project basis is to set up a special account on a limited access node with only that data to be made public available on the node. Figure 2.7-2 indicates examples of data base items that could be shared but typically are not. This type of data base with network access should be a part of the RASSP implementation phase. Recommendation for using standards to implement this environment are:

- Choose CAD, CAM, and CAT tools available on multiple platforms;
- At a minimum provide facilities for binary file transportability;
- To facilitate file sharing between applications, encourage the use on non-proprietary file types such as DOS ASCII or postscript.
- Current popularity would suggest a combination of NFS, Ethernet, and TCP-IP for wide area networking.

2.7.2.4 Other Manufacturing Practices and Interfaces—There are other cost and risk reducers that can be implemented by manufacturing that will benefit the RASSP program. Some of these are indicated in Figure 2.7.2.1-1. Just-in-time procurement of parts reduces the cost of parts inventory storage. In those situations where the volume of parts business is significant to the supplier, qualified suppliers will establish a certified inventory in their own facility and provide inventory storage at their own cost. If the RASSP assembly is highly automated, then parts could be stored directly on the automatic pick and place machines. Assemblies would be inventoried as spares, and only for an agreed upon period beyond the model year upgrade.

Many Honeywell divisions have eliminated or are in the process of eliminating incoming parts inspection. These divisions either buy qualified parts or buy from qualified manufacturers. They then do qualification and reliability testing only at the assembly or box level, and only to the extent required for the specific application. In process testing, even if only to verify the value of parts before insertion, is still performed and is used to statistically monitor the assembly process. Statistical process control (SPC) provides the basis for continuous process improvement. Activity based costing (ABC) is also becoming an accepted practice in manufacturing. It helps identify high cost steps in the manufacturing process and focuses efforts for cost reduction and process improvement.

2.7.2.5 Design/Manufacturing-to-Field Support Interface—Not to be forgotten is the interface from design and manufacturing to and from the field. The ASCEP planning cited at the beginning of this section must include field support considerations. Most important among these is field test. A consistant test strategy for the RASSP signal processor is important and should include field testing. This could present a real challenge because of the potential breadth of applications and military service needs. Field test requirements impose a significant constraint and cannot be ignored. Further, network linkage to the field should be implemented to provide direct access to the product data base. Test reports from the field are a significant input for the continuous process improvement (CPI) process.

2.7.3 Honeywell's Vision for RASSP Manufacturing

The above mentioned cost and risk reducers are most effective when implemented in a manufacturing environment with sufficient volume to maintain a sustained product flow. Since such volumes are rare for military electronics procurements, our preferred site for RASSP manufacturing/production is in a commercial manufacturing setting, a facility capable of producing military quality products in lot sizes as small as one upon demand and intermixed with the ongoing commercial production flow. Demand flow or "pull" is becoming a common comercial practice but would typically not apply to the manufacturing of military electronics. The goal for RASSP production should be in this direction.

Figure 2.7.3-1 attempts to capture Honeywell's vision of a RASSP factory. The key elements for rapid prototype or product development are integrated and sustaining engineering/design, flexible (preferably commercial) manufacturing, and product support departments/organizations operating together in a concurrent product development environment, all served by a community of military and commercial vendors, and all tied together by an information network infrastructure. The model year concept would be an integral part of this product development environment. Design and manufacturing departments could be either co-located or physically separated, but they would work together as a product team fully cooperating, seamlessly functioning, and sharing data bases. Structurally this organization might not look much different from traditional organizations, but functionally it would operate very differently deriving its energy from a participative and empowered organizational culture. Note that more formal practices for treating software will be implemented both within the RASSP factory and within the serving vendor network.

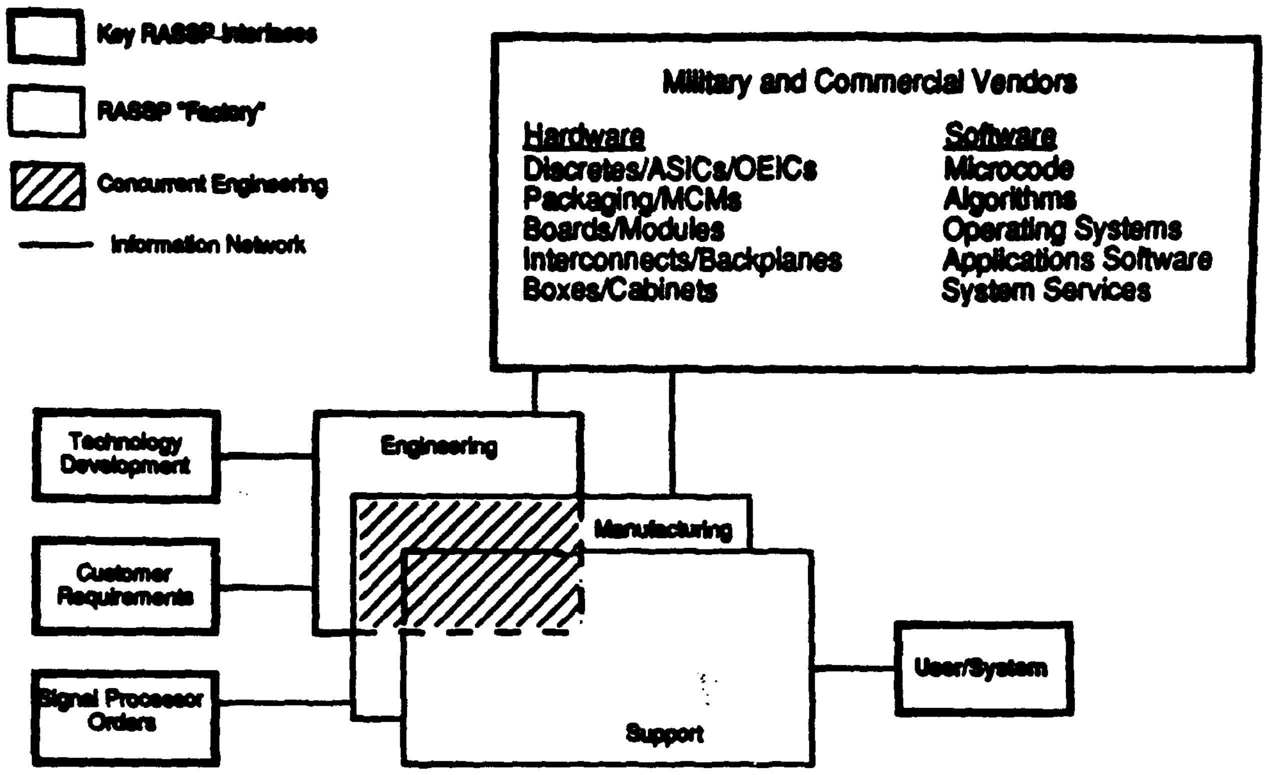


Figure 2.7.3-1. Honeywell Vision of a RASSP Factory

2.7.4 Circuit Card Assembly (CCA) Cost/Complexity Model

One of the benefits of sustained manufacturing is the development of experience-based, empirical metrics for performing cost/performance tradeoffs and for monitoring continuous improvement. An example is the CCA cost/complexity model utilized by one of Honeywell's commercial avionics operations. This model is summarized in Figure 2.7.4-1.

A normalized board unit has been defined and characterized by five factors: (1) 70% component density, (2) 33 pins per square inch node density, (3) 1.3 wires per square inch route density, (4) circuit type, and (5) board type. An actual board unit is calculated as the product of the actual values of these five factors, where CDF, NDF, and RDF are actual/normal ratios, and CTF and BTF have values in the range from 1 to 2. Cost is then calculated as the product of a board unit times historical cost, and cycle time to produce the board is calculated as the product of a board unit times historical cycle time. These tradeoff computations have been programmed as macros into Honeywell's Mentor BoardStations and they can be automatically calculated as part of the design tradeoff process. To be efficient and effective, RASSP design will require these types of metrics and models from manufacturing.

"Normal Board Unit":	- 70% Component Density (CDF)
	- 33 Pins per Sq.In. Node Density (NDF)
	- 1.3 Wires per Sq. In. Route Density (RDF)
	- Circuit Type; Digital (CTF)
	- Board Type; Rigid (BTF)

A Board Unit = (CDF * NDF * RDF * CTF * BTF)

CDF, NDF, RDF: Actual / Normal
CTF: 1 to 2.0
BTF: 1 to 2.0

Cost = (Board Unit * Historical Cost)

Cycle Time = (Board Unit * Historical Cycle Time)

Computations are automated: Macro coded on Mentor BoardStation 8.0

Figure 2.7.4-1. Circuit Card Assembly (CCA) Cost/Complexity Model

Section 3
Important Findings and Conclusions

This RASSP study has sought to achieve several goals:

- Establishment of a RASSP methodology that takes into account state-of-the-art (SOA) and projected development of technology
- Identification of enabling technologies for RASSP
- Projection of the development of the enabling technologies through existing programs and standards
- Identification of the missing and under-developed technologies
- Identification of problems and recommended solutions

The first and second section of the final report is devoted to the baseline RASSP methodology and technologies.

In this section the RASSP methodology and enabling technologies will be presented on the roadmaps. A distinction will be made between the technologies to be monitored and missing technologies that need to be developed. Finally, solutions will be proposed for identified RASSP problems and issues.

The evolution of RASSP is shown in Figure 3-1. The goal of the RASSP implementation phase, which directly follows the study phase, is to establish a RASSP factory, a RASSP product using the RASSP modular architecture, and the RASSP support infrastructure. At completion of the implementation phase, the RASSP infrastructure, product, and factory should be established to the point where further developments will follow without the direct need for additional government funding. RASSP, if implemented properly, will be driven forward by commercial and government market demand.

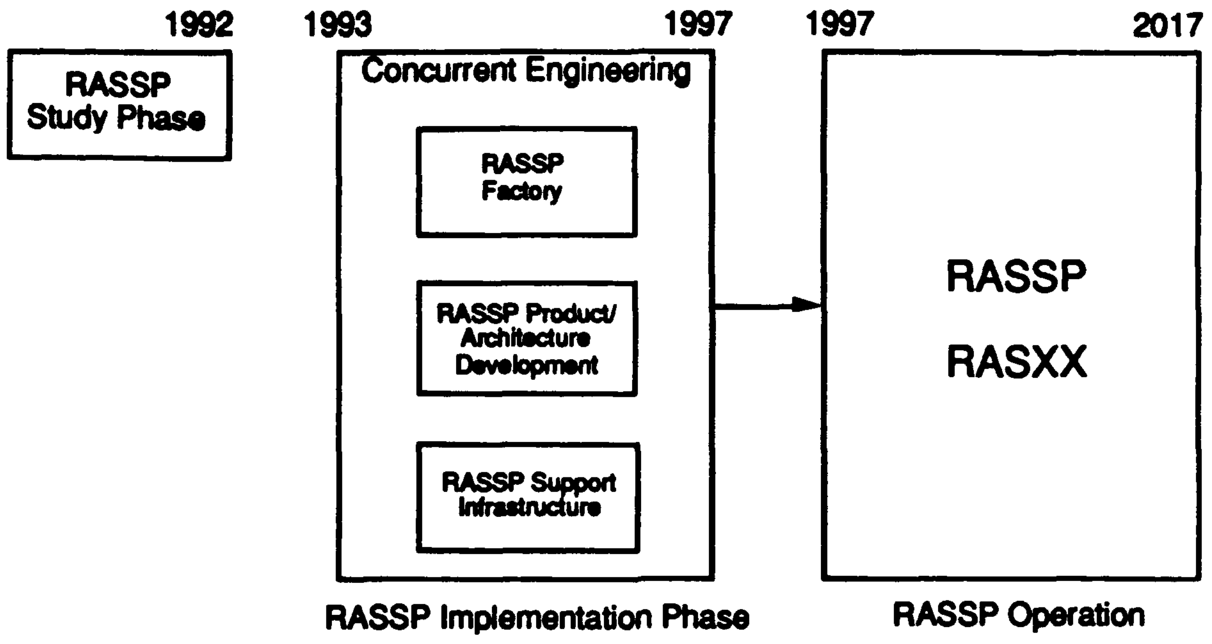
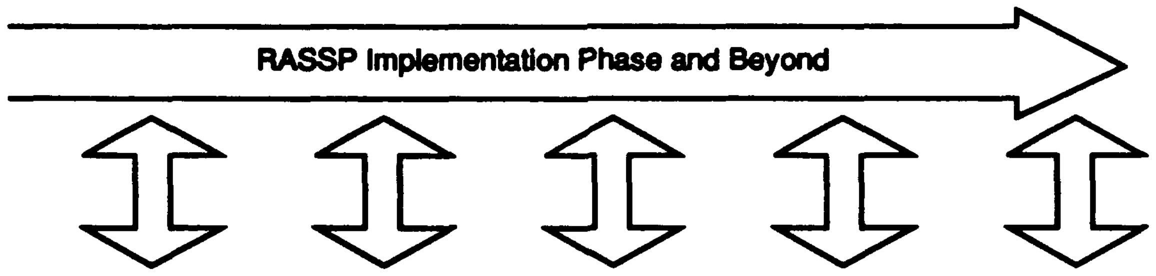


Figure 3-1. RASSP Implementation and Beyond

In addition to following sound business practices enabling full involvement of the commercial sector, as discussed in Section 1, the RASSP concept has to embrace and evolve with on-going standards efforts. As shown in Figure 3-2, these efforts include DoD, joint DoD/industry, and industry standards. To this end, we have identified important standards for all enabling RASSP technologies throughout this report (especially in Section 2).



<u>DoD Standard Activities</u>
- **Process:** CALS, JIAWG, MHDL, SMDs, ADA
- **Product:** MIL-M-38510, MIL-STD-883

<u>Joint DoD/Industry Activities</u>
- **Process:** VHDL, ATLAS/ABET/WAVES, TEDL, AHDL
- **Product:** POSIX, IEEE, 1149.x, FDDI, HIPPI

<u>Industry Standards Activities</u>
- **Process:** IGES/PDES/STEP, CFI, EDIF, IPC, C/PASCAL
- **Product:** IRINC, ARINC 629/659, VME

Figure 3-2. RASSP Implementation Must Embrace Ongoing Standards Development

3.1 System Recommendation

The discussion of candidate systems in Subsection 2.1 presented our findings that ATR ranked number one, the comm system number two, and EW third in our evaluation of their RASSP applicability. ATR was identified as the most challenging target system for RASSP implementation possessing both strength and weaknesses. In this section the ATR is examined as the system of choice for RASSP implementation.

3.1.1 State of the Art ATR System

Honeywell is well-qualified to describe a state-of-the-art ATR system. With over twenty years involvement in the development of automatic target acquisition, tracking, and recognition systems, Honeywell has experienced several generations of algorithms, software, and hardware. These experiences included work under several Government programs: ATSS, PATS, PATS II, ISA, MTAP, SARTC, MSAD, MSFF, LARAA, ALTC, ATR Science, MAIDA, AFID, ASP for self-munition, and Aladdin. In addition, ongoing IR&D is currently focused on a Flexible Real-Time Testbed that is being flight tested. Honeywell has been an active participant in the joint industry/Government sponsored ATRWG since its inception and recently hosted an ATRWG meeting at SRC.

An ATR can clearly be partitioned into an analog front end, digital front end and an embedded processor with mass memory. The analog component includes the sensor(s) and the interfaces with the sensors and with the displays. In addition to interfacing with the analog front end, the digital front end accepts inputs, such as parameters, target models, and digital map data. The digital front end component normally consists of computationally intensive pixel and signal preprocessors. The distinction between the digital front end and embedded processor is not clear cut, and the goal is to move the embedded processing as far forward as possible. Generally, the embedded processor performs numeric and symbolic processing in an ATR system.

Automatic target cueing and tracking are matured technologies available for system insertions. These functions, however, are mission specific; i.e., for certain missions the ATC may need to adjust its parameters in order to detect the required targets. In some instances an ATC will fail. It is important to understand the limits of an ATC and not to over extend its capability.

Commercial hardware for building an ATR system for real-time operation is available. As examples, Datacube Inc. and Imaging Technology Inc. have VME modules that perform most of the necessary image processing functions in real-time. Honeywell has built a real-time flexible testbed (flexbed) that can be configured into an ATR. One of our flexbed systems is currently being flight tested in Boeing's Advanced Avionic Aircraft. Many other architectures for image processing are on the horizon, but most of them are suitable for only commercial use. Military grade processors are still largely missing.

In the area of algorithm development, numerous algorithms have been reported for successfully performing the ATR functions, but they have been tested on limited data in very restricted scenarios. Large scale performance analysis is need. These algorithms, to mention a few, include neural net, genetic search, model-based, and wavelet. The most promising are the model-based ATR algorithms. We see ATR algorithms reaching their final development stage

FLIR has become the primary sensor for ATR systems. FLIR sensors, themselves, have gone through several development stages. The most advanced is the second generation FLIR. Its characteristics include non-interlaced operations, digital output, isotropic sampling, wide dynamic range, and DC restoration/radiometric output. All these will provide unambiguous imagery to an autonomous processor. Synthetic aperture radar (SAR) can also be used in an ATR system, but they are not as commonly used.

Studies have shown that a single sensor will not satisfy ATR needs under all scenarios. Hence multisensor ATRs have been proposed. Other sensors include MMW radar, laser radar, LLLTV, and SAR. Ever expanding functional requirements, such as pre-mission briefing and digital map with INS/GPS, could demand even more inputs. Adding a new sensor could require upgrading all three system components unless such expansion has been anticipated in the original system design. Such forward planning in design is a RASSP goal and our proposed architecture will be particularly amenable to this. More functionality and controls, such as resource allocation, multisensor fusion, and graceful performance degradation can be anticipated.

Processing requirements for an ATR system heavily depend on the selected algorithms. A conventional ATR generally performs region of interest location, connected component analysis and feature extraction, tracking, clutter rejection, and segmentation, target detection or target recognition, and configuration recognition. Based on our experience, we have estimated the processing requirement on each of these functions and computed a total requirement. The result was 1.75 giga-instructions per second (GIPS). Note that GIPS do not correspond to GFLOPS. The conversion from GIPS to GFLOPS depends on the processors used in the system architecture.

Similarly, we have postulated the processing functions for a multisensor ATR (FLIR, and MMW), and a reconnaissance/surveillance ATR. The processing requirements for these functions were estimated and summed to 2.5 GIPS, and 5 GIPS respectively. These processing functions and processing requirements are shown in Figures 3.1-1 and 3.1-2 respectively. The estimated time frame for each of these three ATR types to be matured for production is also shown in the figures. Figure 3.1-2 also includes a list of the additional new sensors and special purpose processors that are needed for the corresponding ATR.

92	93	94	95	96	97	98	99	00	01

ATC

ROI Detection (1.5 GIPS)
Connected Component Analysis (210 MIPS)
Tracking (1 MIPS)
Clutter Rejection (0.2 MIPS)
Segmentation (2 MIPS)
Target Detection (1 MIPS)
Configuration Recognition (1 MIPS)

Multi-Sensor ATR

Target Recognition (540 MIPS)
Hypothesis Generation (0.1 MIPS)
Genetic Search (0.3 MIPS)
LADAR Processing (8 MIPS)
FFT (1 MFLOPS)
MMW Signal Processing (1 MFLOPS)
Fusion Processing (3 MIPS)
Morphology Processing (46 MIPS)
Confidence Accrual (0.5 MIPS)

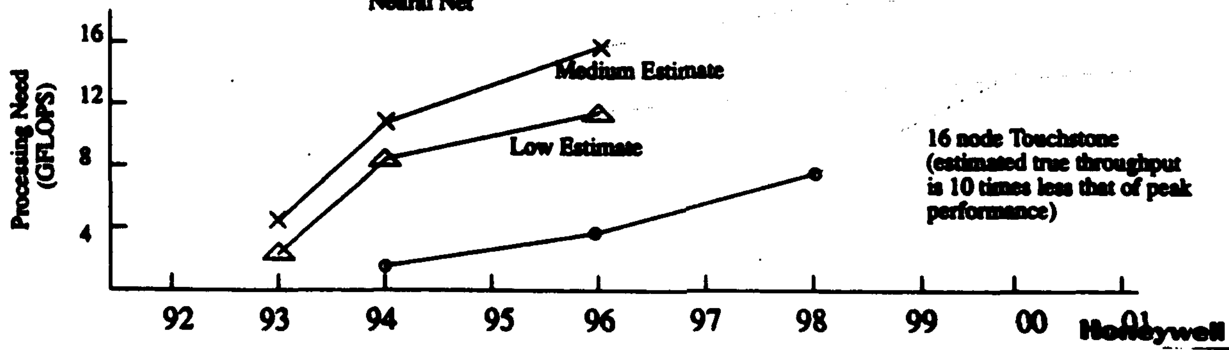
Reconnaissance/Surveillance

Navigation Info. Integration (2 MIPS)
Sensor Info Integration (1 MIPS)
Digital Map Integration (2 GIPS)
Registration (2 MIPS)
Change Detection (10 MIPS)
Image Warping (32 MIPS)
Target Model Retrieval (0.5 MIPS)
Target Identification (1 GIPS)

Figure 3.1-1 ATR processing functions

92	93	94	95	96	97	98	99	00	01

New Sensor (Analog FrontEnd)	Second Gen. FLIR A/D Converter	LADAR, MMW Radar	SAR
Digital Front-End	LUT Convolver I Convolver II Interval Generator MUX Fast GP	FFT Processor Morphology Processor	Warper
Embedded Processor	D/A Converter Frame Buffer	CD ROM List Processor Neural Net	



Honeywell

Figure 3.1-2 ATR processing requirements

3.1.2 Characteristics of ATR System as RASSP Target System

The strength and weakness of the three candidate target systems for RASSP implementation were discussed in Subsection 2.1.3. In this section, we focus on five major deterrents and issues that could hinder specifically an ATR as a target system for RASSP. For each of the deterrents, we also propose recommendation(s) to resolve or reduce the risk. Table 3.1-1 outlines these deterrents/issues and the corresponding recommendations. Detailed discussions follow.

Table 3.1-1. ATR As RASSP Target System

Deterrents / Issues	Solutions / Options
Limited successful operational scenarios.	Start with the limited scenarios; model year upgrade to more scenarios, missions according to the ATR roadmap.
Difficult, high cost performance evaluation, due to lack of database.	Careful experimental design for field data collection. High fidelity IR simulations. More accurate IR signatures on terrain board data.
Almost non-existing standards (Esp. in image format & interfaces).	Select a standard from ATR Working Group, or National Imagery Transmission Format. Define other interfaces and bus standards.
Lack of algorithms for high resolution images.	Support and emphasize model-based, artificial intelligent algorithms.
No military part meets the form, fit, function spec.	Militarized Touchstone, and other high performance image processing modules. Relax military requirements.

Recommended Approaches:	Use FRACTIL and Touchstone as baseline demo programs.
	Increase functionality to ATR as one model year upgrade.
	Upgrade to multi-platform ATR as next model year goal.

The key deterrent is that most potential users do not realize the limited success of the ATR system in finding targets. State-of-the-art ATR only succeeds in some scenarios and missions. Under other conditions, such as a heavily cluttered scene, occluded target, or in the presence of counter-measures, an ATR probably fails its target finding mission. Users must understand the capabilities of current ATR systems and not attempt operations beyond these limits.

We recommend that the RASSP program accept the current ATR capability and establish it as the baseline for RASSP implementation. The performance improvements and increases in ATR functionality can be treated as model year upgrades. The baseline model can be an ATC. A proposed road map for ATR systems plus various upgrades are discussed in next section.

Industry's lack of ATR standards is another deterrent. Limited existing systems and no production base has each potential ATR competitor engaged in their own proprietary development efforts. For RASSP to successfully produce an ATR system, the government, in cooperation with industry, must select and establish ATR standards. DARPA has been sponsoring the ATRWG. A committee can be established to address standard issues. The interface between the sensor and the digital front end is utmost important. Other standards such as bus, image format, and pixel resolution should also be established. RASSP then must enforce these established standards. They would, in fact, be one of the RASSP program's outputs.

The third deterrent is that current hardware does not meet military form, fit, and function requirements for an ATR. There are two reasons for this. First, the ATR functions require special purpose hardware, such as convolver modules, to achieve real-time operations. Second, a preferred system architecture for an ATR does not exist. Individual mil-spec hardware components exist, but these components cannot be readily combined into an acceptable ATR system.

We recommend two parallel approaches to resolve this deterrent. First, the government needs to find a less restrictive approach for meeting the military's special hardware requirements. Commercial hardware that performs the required ATR functions exists but is not qualified for use in military systems. Qualifying the systems rather than their parts may be part of the solution. Qualifying manufacturers rather than their piece parts may be another element of the solution.

The parallel approach is to establish a standard ATR system architecture. Then special hardware modules can be built to perform the ATR functions. Since these modules would implement standard interfaces, they could be readily interchangeable and upgradable. We have studied a system architecture that is suitable for building an ATR system.

Another difficulty that an ATR faces is in performance evaluation. Conducting field tests is the preferred method for evaluating an ATR's performance, but this method is very costly and often suffers from missing data. Collecting the necessary ground reference data is often not well-coordinated. Data, such as the aircraft flight dynamics, exact target location relative to the platform at any moment in time, sensor viewing geometry, and weather conditions, are essential for performance evaluations. Often, operator interference, e.g., changing a gain to best satisfy the operator's vision, can cause poor ATR performance.

We recommend building a library of field data for performance evaluation. This data base should include the ground reference data necessary for evaluation purposes. Collaboration with other government agencies is needed. RASSP would not be the appropriate program to fund a field data collection effort, but WL, for instance, has planned to conduct a data collection effort under the FRACTIL program.

In addition to actual field data, the library can be supplemented with simulated data. The state of the art high fidelity IR simulation, however, does not provide the necessary realism in the simulated imagery. Many efforts have been devoted to this area. Trustworthy simulation is foreseeable in this decade. Another addition to the field data in the data library could be the terrain board data. NVEOD has devoted many efforts in establishing a terrain board where imagery resembling that of IR can be collected.

The last deterrent is that SOA ATR algorithms do not deal with high resolution imagery generated by sensors such as the second generation FLIR. Lack of data, and insufficient test and evaluation are the causes of this deterrent. Some advanced algorithms have demonstrated promise, but due to their complexity, the likelihood of implementing them in real time operation is very small. Moreover, the processing time for these algorithms is so long that extensive tests on them are prohibitive on a workstation.

Our recommendation to address this issue is to continue support for advance algorithm development. Model based vision, as the SOA approach, should be emphasized in RASSP. As the processor performance increases, more sophisticated algorithms giving better system capabilities can be embedded. ATR is an excellent example. Previously, ATR performed line-based processing; now large window based processing is common. Statistical classifiers were heavily used in the early ATRs; the computation-intensive model-based classifiers have captured most attention now. Genetic algorithms, morphology, and neural networks for searching and pattern matching are foreseeable.

In summary, we identified five deterrents for ATR as the RASSP target system; they are limited successful operation scenarios, no commonly accepted standards, lack of military qualified hardware, difficult and costly performance evaluation, and the lack of sophisticated real-time algorithms. RASSP should address the first three in the implementation phase and support actions for resolving the last two.

3.1.3 ATR Model Year Upgrade Paths

As previously stated, one of the ATR's strengths is its numerous opportunities for model year upgrading. I n this section, we suggest and describe three model year upgrade options from a baseline ATR system. Each upgrade increases the performance, the number of users, or the functionality of the ATR. Figure 3.1-3 shows the three upgrade options, and the additional elements that are required by each upgrade.

The baseline ATR is an ATC/ATR for air support purposes. The three upgrade options are, more platforms, precision strike, and reconnaissance/surveillance. The proposed baseline ATC/ATR system is for air support. In this application, the ATR system scans the battle field area, detecting and locating all the potential targets with minimal false detections. The final decision and hand off to fire control is made by an operator. As such, this ATR system is often called automatic target cuer (ATC) which serves as an aid to the operator. This baseline ATR will require a second generation FLIR sensor, proper interfaces, and high speed special and general-purpose hardware. To optimize the performance, pre-mission briefing about the weather conditions, and expected target types are essential to know a priori. The targets to be detected are tactical targets such as tanks, APCs, howitzers, trucks, etc. This ATC will operate both day and night under good weather conditions looking at a slant angle from air to ground.

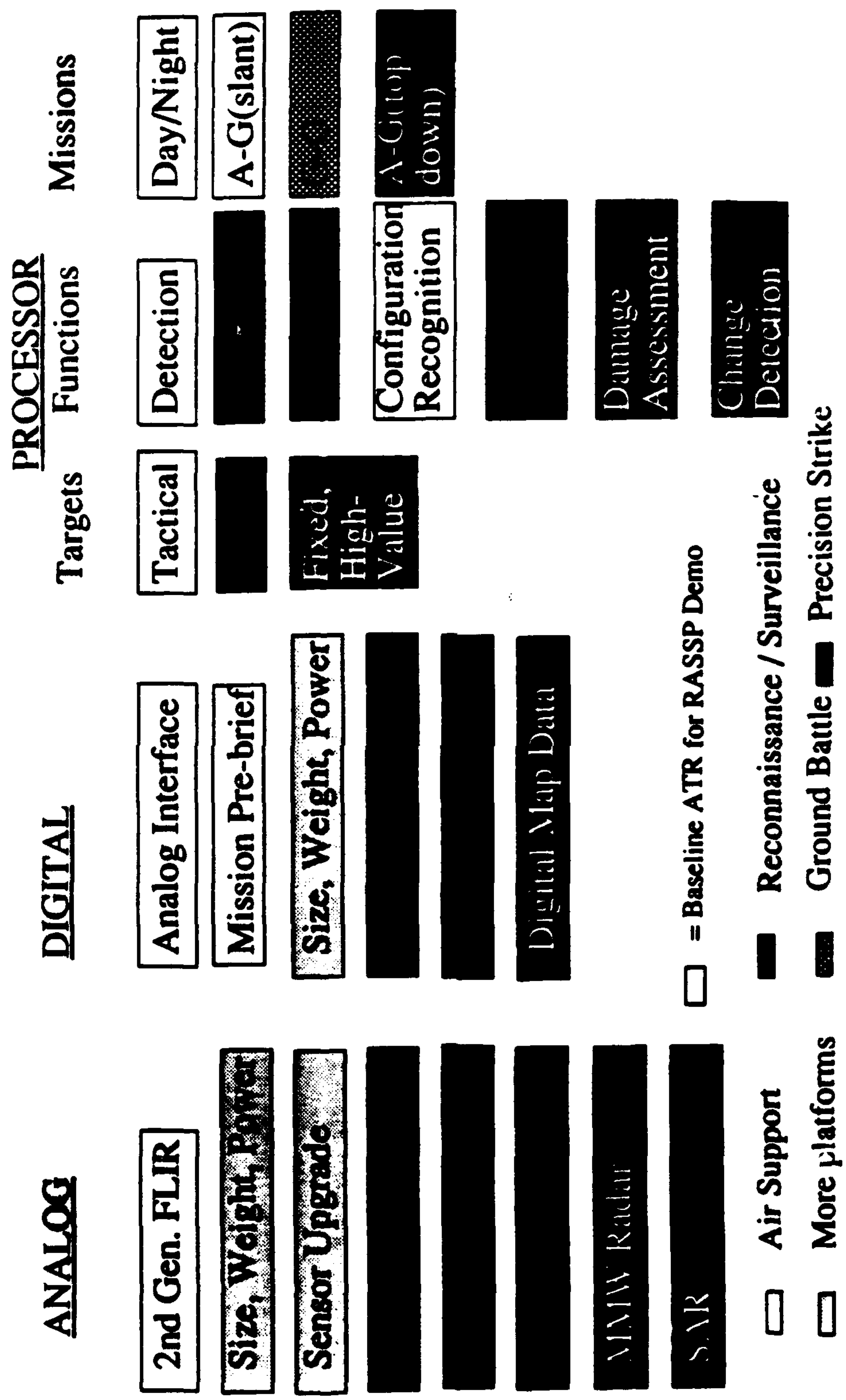


Figure 3.1-3 ATR Model-Year Upgrade Paths

From this baseline, the ATC can easily be upgraded for installation into multiple platforms. The upgrades will be mainly, size, weight, and power reductions of the ATC system. This will allow a better form and fit into the specific spaces of various platforms. Sensor upgrading is also postulated. This will enable the sensor to be mounted on a ground vehicle such as a tank, changing it to a ground-to-ground operational ATR.

A major upgrade from the baseline ATR system is required to enhance the ATR function to precision strike capability. Under this mission, the capabilities for accurately locating, identifying, and killing high-value time sensitive ground targets are needed. To achieve these capabilities, the ATR system must use a lot of other information and resources. Multiple sensors including, ranging, laser sensing, and low light-level TV should be used in order to collect more data for making accurate identification. Additional mission and intelligence data about the targets increases the probability of success. If available, GPS and navigation data from the platform also help in finding the targets. Multisensor processing, that which coherently combines and processes the data from different sensors, is needed. In addition, the detection function in the baseline version needs to be upgraded to the identification level, which indicates the specific model of the target type.

Another model year upgrade option is to augment the ATR application to surveillance and reconnaissance. In this option new sensors are also added; they include the MMW radar and SAR. To more accurately locate targets, digital map information is desirable. The viewing geometry will change to a top down or high slant angle at high altitude. Functionally, the ATR needs to additionally perform damage assessment, chance detection, information fusion, and report generation.

3.1.4 ATR Road Map and Supporting Programs

The preceding subsection outlined three model year upgrade options. This subsection provides a bigger picture of the ATR roadmap. It attempts to lay out the projected progress of the ATR technologies and applications, both in the military and commercial worlds. A list of ATR related government programs is also included.

Figure 3.1-4 shows the forecasted roadmap for ATR technology. The different stages of ATR capabilities are shown as shaded boxes. Specifically, the baseline and three options of upgrades correspond to the first five rows. Combination of the multi-sensor ATR and multi-mission ATR yields the precision strike capability referred to in the previous section. The progression of each stage is indicated by the heavy black arrows.

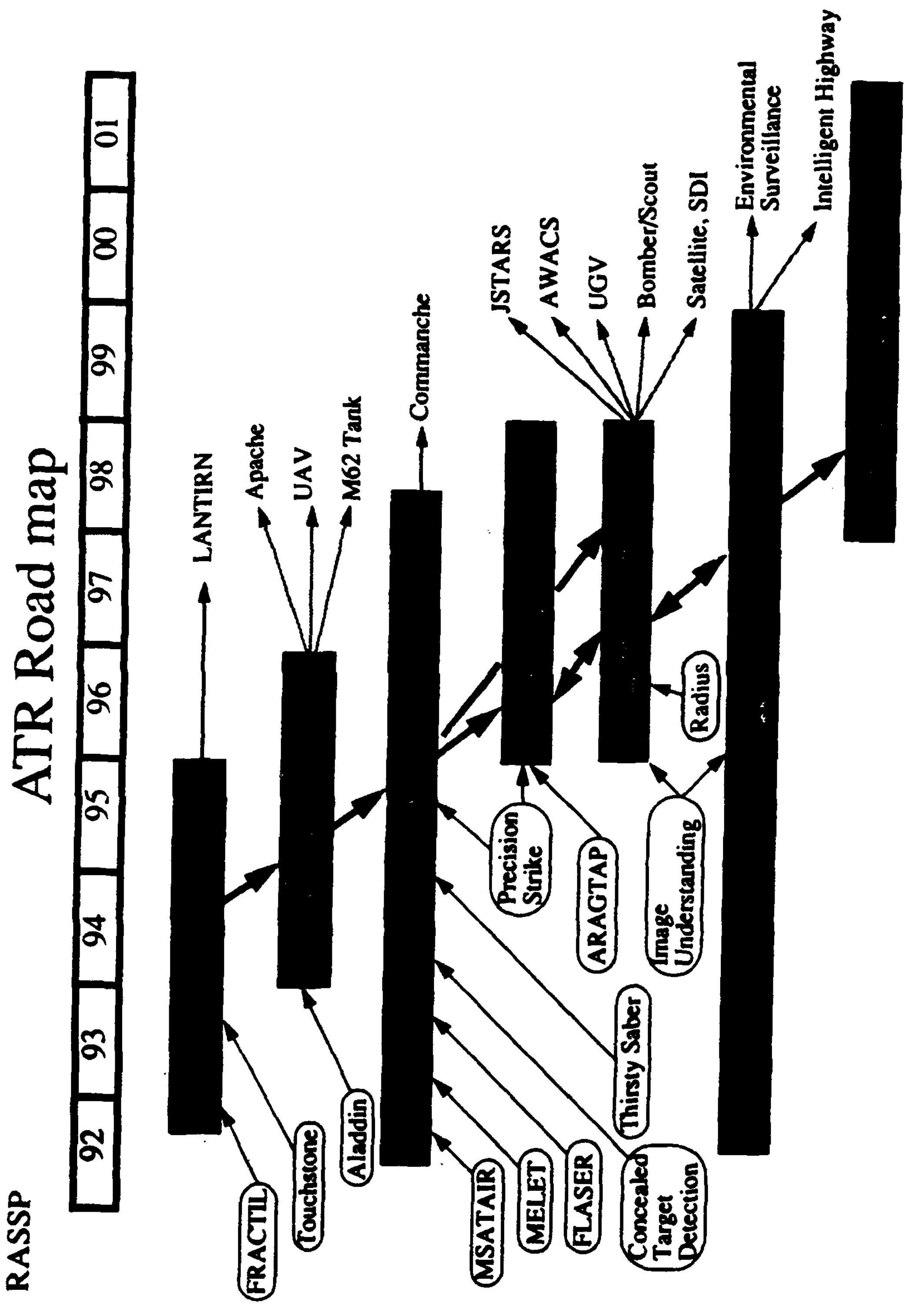


Figure 3.1-4 ATR Road Map

The applications or platforms that employ the ATRs are indicated by the shaded arrows emerging from the end of the boxes. These applications and platforms are by no means exhaustive. Many others can easily find the appropriate ATR useful, especially in the commercial arena. In the same figure we also indicate some government-funded programs that contribute to these ATR technologies. These programs were drawn inside the oval box and pointed to the corresponding ATR technologies. Also, provided in Table 3.1-2 is a description of the objective, and the technical representative for each of the programs. In addition, if known to us, is included the prime contractor's contact person for the program.

Table 3.1-2 ATR System Programs

Program Name	Objective	Gov't COTR	Prime Contractor
FLIR Auto Cuer Technology Insertion into LANTIRN (FRACTIL)	To incorporate an ATC into the LANTIRN targeting pod.	Higgenbotham Lantirn SPO, WL (513) 255-2222 Lloyd Clark AARA-1, WL (513) 255-1115	Field data collection started RFP for ATC Evaluation in procurement
Touchstone	To produce a militarized version of the commercial Intel Paragon processor	Gil Weigand DARPA (703) 696-2227	Ed Collum SASSO, Honeywell (813) 539-5397
Aladdin	To produce a soup can sized, 1 GFLOP processor	John Hodapp NVEOD (703) 664-5207	Texas Instrument Alliant Tech System
MELET	To demonstrate active techniques of laser radar for IFF, low observable target detection and BDA	Hank Lapp, Cpt. Matt Rotondaro AARI, WL	Northrup TI
RADIUS	To increase the productivity of the image analysis with tools, utilities, and use of site models	Don Gerson ORD, CIA (703) 351-2708	Randy Onishi Hughes Aircraft (310) 616-0733
FLASER	To develop the methodology and tools required for the design of an EO Passive/active multispectral A-G ATR system	Hank Lapp Jan Servaites AARI, WL (513) 255-5922	To be procured
ARAGTAP	To design a SAR model based ATR	Steve Sawtelle AARA, WL	Dave Morgenthaler Martin Marietta (303) 977-4200
Concealed Target Detection	To flight demonstrate technology capable of searching wide areas for CMT concealed under foliage / camouflage	R. Davis AFSC/XTTA DSN 858-5065 Jean F. Roman AARM, WL (513) 255-6427	Approaching end of first phase; Second phase planned
ARTEMIS Precision Strike Demo	To develop demonstrate affordable capability to precisely deliver conventional weapons against time-critical fixed and mobile targets	Umbrella program jointly funded by Air Force, Navy, Army, and DARPA	Planned procurement
Image Understanding	To develop image understanding, analysis techniques for robotic, vision, and recognition applications	Umbrella program funded by DARPA	Many defense companies and universities
MSATAIR	To develop a multi-sensor (FLIR, LASER radar, MMW) ATR	NVEOD	

We conclude that ATR is of interest to many users. A clear path and many applications for an ATR system are projected and shown in the road map. We anticipate that ATR technology will mature and flourish by the end of this decade. It is therefore an excellent candidate for RASSP implementation.

3.2 System Design

The system design task emphasized both process development and product insertion opportunities. The evolution of the system architecture, packaging, software, memory, and interconnect should track the rapid increase in system insertion requirements . The system design tools should be developed to handle the system requirements and technologies as they evolve. The roadmap for the RASSP insertions to the fast-evolving ATR system is presented in Figures 3.1-1 and 3.1-2, and ATR needs are described in the previous section. The challenge for the RASSP/ATR architecture is very fast advancement of the hardware (memory, interconnects and co-processors).

The fast increasing performance of hardware leads to significant improvements in software, as shown in Figure 3.2-1. For example, as the memory sixe and the interconnect bandwidth grow the fully object-oriented software becomes feasible.

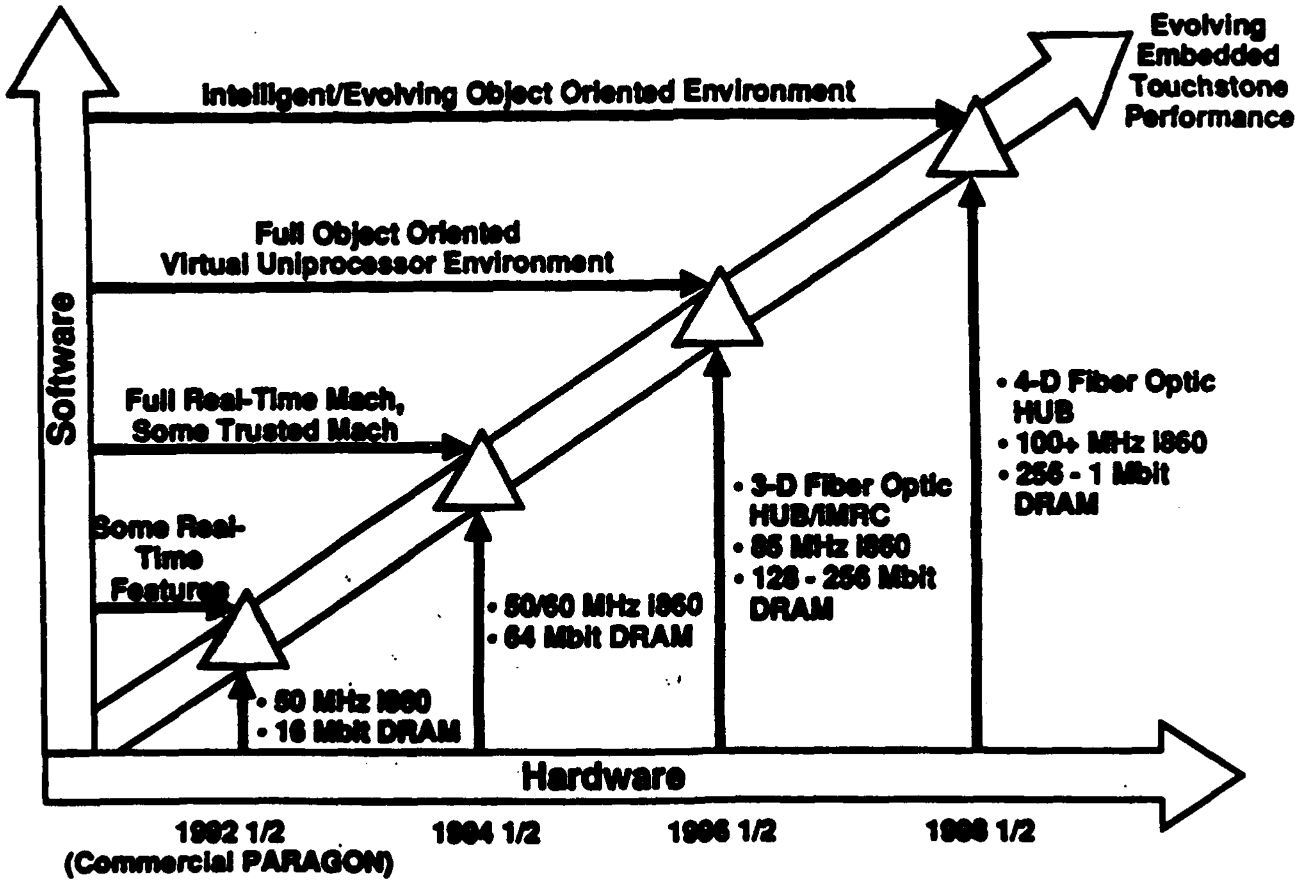

Evolving Embedded Touchstone Performance
Intelligent/Evolving Object Oriented Environment
Full Object Oriented Virtual Uniprocessor Environment
Software
Full Real-Time Mach, Some Trusted Mach
Some Real-Time Features
• 50 MHz I860
• 16 Mbit DRAM
• 50/60 MHz I860
• 64 Mbit DRAM
• 3-D Fiber Optic HUB/IMRC
• 85 MHz I860
• 128 - 256 Mbit DRAM
• 4-D Fiber Optic HUB
• 100+ MHz I860
• 256 - 1 Mbit DRAM
Hardware
1992 1/2 (Commercial PARAGON)
1994 1/2
1996 1/2
1998 1/2

Figure 3.2-1. Evolving Software and Hardware Technologies Ensure the Long-Term Viability of the Embedded Touchstone

A solution to these problems is the RASSP architecture which will accommodate the high performance co-processors as they mature. RASSP architecture will have a flexible, open interface for future upgrades (model years). For example, the signal processor front end will be different for different sensors and varying scenarios. A possible solution is offered by the Touchstone architecture (see Figure 3.2-2) that is based on the standard Intel Paragon interfaces (e.g., i860 microprocessor, data bus, address bus, and memory). The daughter board can be inserted as needed for preprocessing or graphics processing applications. The hardware and software format is uniform and stable to simplify the interfaces (e.g., the daughter board footprint and I/O pattern stays the same, the software operating system has "hooks" for future vector co-processors and application-specific processors).

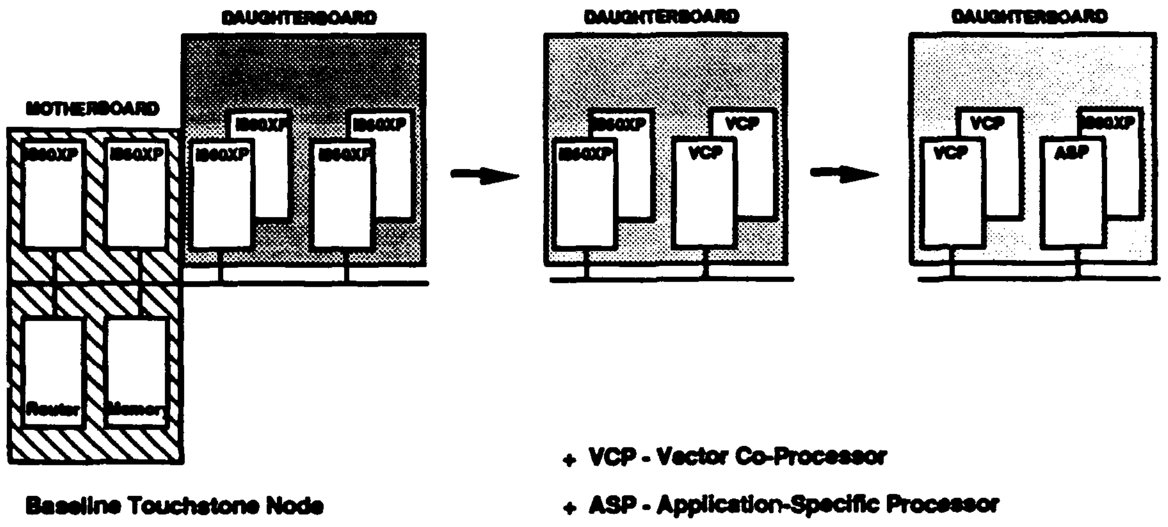


Figure 3.2-2. Next Generation Avionics Processors Potential Performance Progression

The second set of issues for the system design are the tools used for different steps of the design as listed in Table 3.2-1. The areas of particular concerns are documentation, requirement tracking, trade-off analysis, system level modeling/simulation, system hardware/software partitioning, and system test generation.

The documentation problems have to do with the existing standard 2167A that is old and hard to integrate with the new design process. The new documentation requirements should be established or derived from the old practices. Similar recommendations apply to the requirement tracking. Possibly the government program, "Continuous Electronic Enhancement Using Simulatable Specifications," (CEENSS) may facilitate the change.

The trade-off analysis of performance, cost, size, power, and reliability is handicapped by lack of conventional metrics representation, or libraries. The wide variety of existing tools like Honeywell Automatic Trade-off Tool (AToT), Mentor/TI Design for Manufacturability, Multichip System Design Advisor, and Architecture Design and Assessment System need to be integrated around standard metrics and reuse libraries.

System-level modeling and simulation suffer from very slow development/simulation acerbated by immature modeling standards and poor analog/digital simulators. The recommended solution is to introduce behavioral accelerators (a testbed that is a combination of simulator/emulator/hardware-in-the-loop might prove to be the only solution) and to develop analog/digital simulators. The successful practices should be captured in a "System Modeling Handbook."

The system hardware/software partitioning would benefit from better translators and interfaces between CASE and EDA tools. These, in turn, will be facilitated by the development of common high-level representation of both hardware and software. In particular, the signal processing hardware/software trade-off will involve some microcode rapid-prototyping. Therefore, a high fidelity system level modeling/simulator will be necessary for RASSP.

Table 3.2-1. System Architecture Development Issues

Technology	*SOA*	*Programs*	*Standards*	*Issues*	*Solutions*
System Architecture					
• System Analysis	SES Workbench ADAS, Statemate, Ascent Logic RDD, SPW, Synopsys VHDL System Simulator; IR&D	GPD/CDG	VHDL, CFI	Tools poorly integrated, narrow focus	EDA vendors move towards CFI
• Documentation	2167A requirements generated from CASE tools	CEENSS	2167A	Current standards hard to integrate with new design process	Establish/change new documentation requirements
• Requirements Tracking	Ascent Logic RDD, Honeywell ARRTS	CEENSS		Tools tend to be stand-alone; little formalism	Establish conventions in VHDL/documentations
• Trade Off Analysis	Internal tools; Honeywell's AToT, Mentor/TI DFM, MSDA, ADAS	ASEM	VHDL	No conventional metrics, representations, or libraries	Establish reuse libraries with standard metrics; integrated trade off tools
• Formal Verification	CLSI Paris, Vertex Boolean Verifier, IR&D		VHDL	Still very immature; existing tools have limited focus	Standard subset; domain specific verification
• System Level Modeling/Simulation	VHDL models, Acceleration (Zycad, IKOS), GLSS	F-22 Protocol, GPD/CDG	VHDL, IEEE 1164	Speed, speed and speed; lack of modeling standards; poor analog/digital simulators	"System Modeling Handbook," behavioral accelerators; develop a/d simulators
• System Hardware/ Software Partitioning	System analysis tools; manual	CEENSS	VHDL, Ada	No good link between CASE and EDA tools	Develop common high level representation
• System Test Generation	System Test Synthesis, Synopsys Test Compiler	ASEM	IEEE 1149	Tests developed from relatively low level (RTL) description	Test development from requirements, test decomposition/tracking

ADAS	Architecture Design and Assessment System	MSDA	Multichip System Design Advisor
RDD	Requirements Driven Design	DFM	Design for Manufacturability
ARTTS	Avionics Requirements to Test Tracking System	GLSS	Gate Level System Simulation
CEENSS	Continuous Electronic Enhancement using Simulatable Specifications		
ASEM	Application Specific Electronic Module		

The final system-level problem is the test strategy/plan generation. It will be discussed in more detail in Section 3.5.

The other system design tool issues include:

- Discontinuation of a design tool;
- Consistent and concurrent feedback path from hardware, software, manufacturing/ production, testing to system design;
- Standard description of human interface.

As discussed above, a simulation/emulation/hardware test-bed is a likely candidate for the system design environment (see Figure 3.2-3).

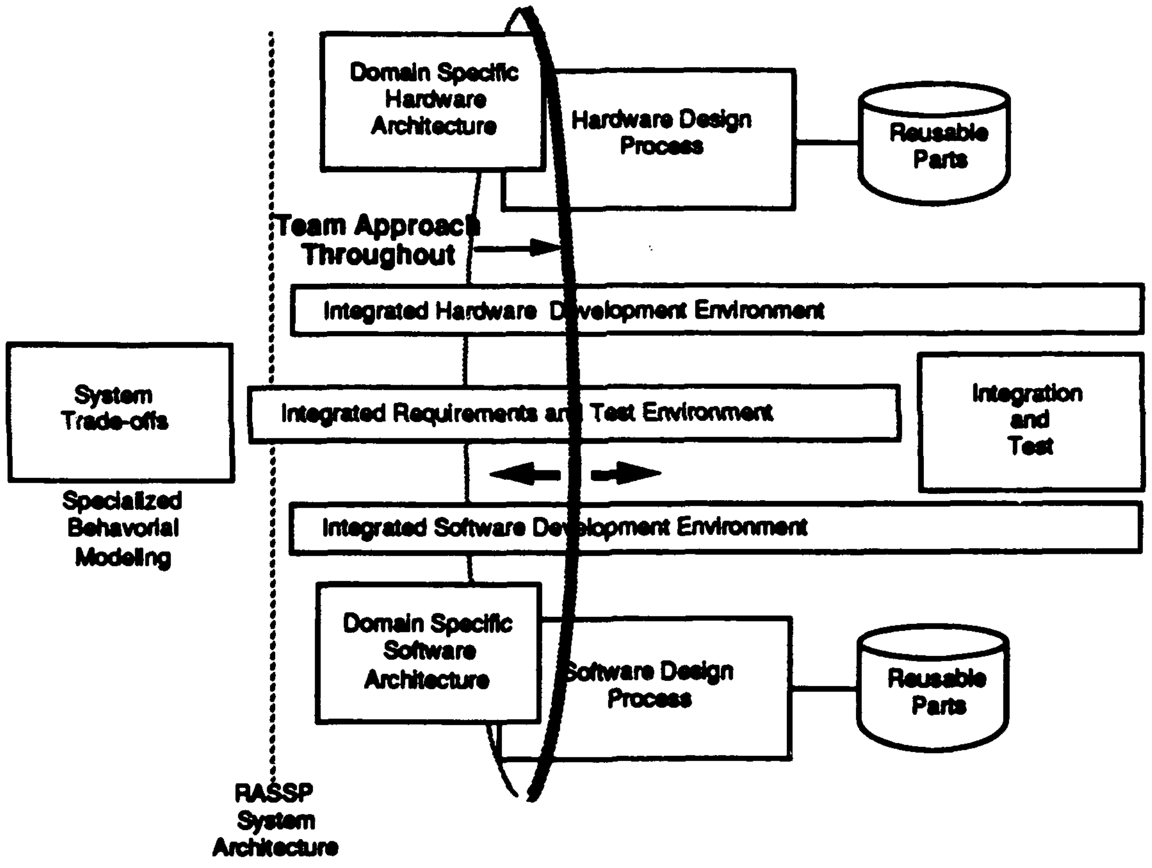

Domain Specific Hardware Architecture
Hardware Design Process
Reusable Parts
Team Approach Throughout
Integrated Hardware Development Environment
System Trade-offs
Integrated Requirements and Test Environment
Integration and Test
Specialized Behavorial Modeling
Integrated Software Development Environment
Domain Specific Software Architecture
Software Design Process
Reusable Parts
RASSP System Architecture

Figure 3.2-3. RASSP Design Process

Such a test-bed will offer the following challenges:

- Continuous evolution of testbed
- Full time support personnel
- instability of testbed by introduction of new tools/technologies
- Non-recurring engineering expenses.

Whereas all of the above issues need to be addressed by the RASSP program, there are some areas that require tracking of progress only. The example is the system analysis where tool vendors move towards common file interchange (CFI) standard. Similarly, the formal verification tools are introduced that address domain specific verification that might be sufficient for RASSP.

3.3 Hardware Design

The hardware design and hardware technologies, i.e., circuit and assembly, MCM, ASIC, are better defined and more mature than the system design issues. Therefore, we do not consider the hardware technology as an issue for RASSP (although it will be imperative to take advantage of new technologies such as GaAs ASIC, photonic interconnects, and new packaging being developed under ASEM programs under DARPA). The success of RASSP will depend on the new architecture technology and how open it will be for the new hardware, software, and manufacturing technologies.

The main issue in the hardware design area pertains to the design tools/methodology and hardware development life cycle. The hardware design development issues and solutions are presented in Table 3.3-1. The most critical needs are in the area of detailed hardware design, PCB design/layout, and MCM design/layout. The tools for the detailed hardware design are poorly integrated.

A solution is to force the tool vendors to move to a common format, i.e., CFI. The challenge of PCB and MCM design is the same—high frequency signals. The tools for simulation of electromagnetic characteristics (i.e., characteristic impedance, reflections, cross-coupling) of the interconnects need to be developed and supported by a vendor. The tool should include trade-off capabilities, for example, optimization for speed, or area, or power.

There are several areas in the hardware design process where progress should only be monitored, because of commercially-driven markets. For example, most of the PLD/FPGA/gate array design need to be converted to ASIC for higher performance. The conversion should be vendor independent. Another area is the ASIC test compilation that suffers from lack of link between software and hardware development. A common, high-level representation will solve this issue. The affordability of ASIC drives the automatic test pattern generation area. However, the automatically generated test vectors are not easily transferred to the system-level representation or to Manufacturability. Again, the trend is to have common representation of test vectors throughout the product development cycle. Similarly, the formal verification and documentation will benefit from standardization, at least in a specific domain. To solve all the problems, an industry wide standard should be encouraged.

Table 3.3-1. Hardware Design Development Issues

Technology	*SOA*	*Programs*	*Standards*	*Issues*	*Solutions*
Hardware Design					
• Detailed hardware design - Simulation - Synthesizers	SES Workbench ADAS, Statemate, Ascent Logic RDD, SPW Synopsys VHDL; IR&D Microsim/Design Center, Mentor Graphics/System-1076, Quickturn, Seep (S25K), Cadence Design Systems/Analog Artist Switched Capacitor Design System Cadence Design Systems, Compass Design Automation, Dazix/Titan, LSI Logic/System 1076, Racal-Redac, Synopsys/Design Compiler, Design Analyzer, Vantage Analysis Systems, Viewlogic	GPD/CDG	VHDL, CFI, Schematics, EDIF, Verilog, Dazix, NDL, Netlist	Tools poorly integrated, narrow focus	EDA vendors move towards CFI
• PLD/FPGA/Gate Array Design	Actel Corp.(ACT 1/40 MHz), Altera Corp. (MAX 5000/100 MHz), AT&T Microelectronics (ATT3000/85 MHz), Intel Corp. (iSCXXX/50 MHz), Lattice Semiconductor Corp. (22V8 /100 MHz), National Semiconductor (GAL/62.5 MHz), QuickLogic Corp. (PASIC 1/100 MHz), Xilinx Inc. (XC4000/70 MHz) Applied Microcircuits (Q20000/1.25 GHz) LSI Logic Corp. (LCA 200K/>100 MHz), Motorola Inc. (MCA 3 ETL/2 GHz) National Semi-conductor Corp. (NGM/2.5 GHz) Texas Instruments Inc. (TGB1000/>200 MHz) Vitesse Semi-conductor Corp. (FX Series/ 1 GHz)		Truth table, Boolean equations JEDEC, Palasm	Transition from functional PLD/FPGA to ASIC	Vendor independent conversion to ASICs for significant cost reductions over the life of a product.
• ASIC Test Compilers	Compass Design Automation (Test Assistant/$60K), Dazix (Testsyn /$25K), GEC-Plessey Semiconductor (Gatemap/$25K), Philips Electronic Design and Tools, (Lasar/m/$30K), Racal-Redac (Sherpa's Test Synthesis/$54K), Sunrise Test Systems (Testgen/$95K) Synopsys (Test Compiler/$40K) Teradyne EDA (Frendship Synthesis/$75K)	CEENSS	VHDL, Ada, Schematics, EDIF, Verilog, Dazix, NDL, Netlist	No good link between CASE and EDA tools Performance impact zero delay scan latches on critical paths	Develop common high level representation
• ASIC ATPG	Augen Software (Augen/$8K), Atlas Software (Atlas TPG/$100K), AT&T (Gentest/$150K), Compass Design Automation (Scan Test ATVG and Star/$40K), Dazix (Plasma/$120K), LSI Logic (SATPG/included with test builder) Philips Electronic Design and Tools (Panther/$30K) Racal-Redac (Intelligen/$127K), Sunrise Test Systems (Testgen/included with test-logic insertion tool)	ASEM	IEEE 1149	Translation of test vectors to the next high level in the hierarchy - eventually to manufacturing.	Common test vector representation throughout development to production ATE.
• PCB Design/Layout	Racal-Redac/Visula ($75K), High Performance Engineering $80K, Dazix/MCM Engineer, Mentor Graphics/MCM Station, AutoTherm, Quad Design Technology/3D Field Solver, CSA International/PG-Plane			High-Frequency Design Analysis	High-Frequency Design Analysis Tool Development
• MCM Design/Layout	Pacific Numerix/PCB Explorer, SolderSim Cadence Design Systems/Allegro CSD Mentor Graphics/ Board Station 800 ($125K) Quicksim, Quickpath, Crosstalk Tool Kit (Quad Design) Pre-route Delay Quantifier (Quad Design) Harris Sitecard	ASEM, MSDA		High-Frequency Design Analysis	High-Frequency Design Analysis Tool Development/Design Tradeoff Tool Development
• Formal Verification	CLSI Parts, Vertex Boolean Verifier, internal research		VHDL	Still very immature; existing tools have limited focus	Standard subset; domain specific verification
• Documentation	2167A requirements generated from CASE tools	CEENSS	2167A	Current standards hard to integrate with new design process	Establish/change new documentation requirements

ADAS	Architecture Design and Assessment System	MSDA	Multichip System Design Advisor
RDD	Requirements Driven Design	DFM	Design for Manufacturability
ARTTS	Avionics Requirements to Test Tracking System	GLSS	Gate Level System Simulation
CEENSS	Continuous Electronic Enhancement using Simulatable Specifications		
ASEM	Application Specific Electronic Module		

The issues of the hardware development life cycle are similar to the system development life cycle; keeping up with the tool evolution, expense of support personnel and continuous training, and phasing out outdated tools. The concurrent interface between hardware development and system and manufacturing development needs to be maintained, as discussed in Subsections 2.7 and 3.7.

The roadmap for the hardware design and system design tool development is shown in Table 3.3-2. The progression of the technology does not include the solutions to the critical issues that need to be introduced, supported, and accelerated by RASSP.

Table 3.3-2. Roadmap for Hardware Design Developmen

	Now	*1993*	*1994*	*1995*	*1996*
System	Stand-alone system analysis tools	Initial commercial integration of CASE/EDA tools; metric definition	Behavioral acceleration; prototype trade off tools based on reuse libraries	Fully integrated analysis and specification tools; standard VHDL conventions for various subsets; new documentation standards	Formal verificaiton and system analysis tools based on standard conventions, including trade off tools
Hardware	RTL Synthesis, commercial reuse libraries (no standards except use of VHDL)	No significant changes	Links between signal integrity analysis and place/route tools; prototype behavioral synthesis	VHDL Analog/Digital simulators, reuse libraries based on standard VHDL conventions	Domain specific behavioral synthesis tools for signal processing

3.4 Software Design

The challenges of RASSP are particularly demanding of the software design:

- Rapid prototyping/upgrade;
- Affordability;
- Very high performance in real-time.

Therefore, new methodology and new tools for software design are strongly recommended. The new methodology for software reuse is described in Subsection 4.1 and is a key to affordable RASSP. The software development technologies to be developed are listed in Table 2.4-1 and 2.4-2 of Subsection 2.4.

The software design issues critical to RASSP success are lack of architecture specification languages and domain specific tools for signal processing. The ongoing DARPA program Domain Specific Software Architecture (DSSA) addresses these two issues for different domains, for example, control and navigation. DSSA will provide a methodology for RASSP software design, however, the architecture specification language and signal processing domain specific tools will need to be developed under RASSP follow-on.

The additional areas that will need monitoring or CASE tool acquisition and/or new standard encouragement are: Domain analysis, architecture analysis/simulation, validation and verification/test, programming languages, embedded operating systems, and microcode development. The existing tools for the architecture analysis/simulation saturate quickly for complex software. The use of the emerging hierarchical tools and/or partitioning of software architecture into smaller blocks is recommended as the RASSP approach. The issues of V and V/test are loose integration with simulation oriented environment and weak traceability to the top level requirements and rationale. RASSP should pursue more domain specific test generation that will encourage/enable more integration and automatic synthesis. The domain analysis is inefficient because of little automation of capture and organization of domain models. The "smart" tools should be used wherever possible. The programming language standardization is another recommendation for RASSP. The proliferation of many languages will increase cost of upgrades, jeopardize the robust, real-time secure performance and degrade interoperability of different modules of RASSP. The embedded real-time operating system is under development, however, its fault-tolerance is still in the basic research stage. A slight redirection and acceleration of the research in loosely-coupled, fault tolerant operating system for the signal processing domain is necessary. Finally, the automatically generated microcode is only slightly less compact than human-written one. Again, the automatic microcode generation development should be redirected to standard signal processor architecture.

The RASSP software technology progression is shown in Table 3.4-1. The first row represents the operating system progression. The date of the RASSP compatible, fault tolerant and real time operating system will be 1995 or 1998 depending on the pace of the ongoing research. The evolution of the fully automatic microcode compiler should be completed around 1994-6. Finally, the last row reflects the progress in the area of generation of microcode from algorithm. The 1995-6 timeframe is feasible if RASSP funds DSSA-type activities for signal processing.

Table 3.4-1. RASSP Software Technology Progression

now	1993-4	1994-5	1994-6	1995-8
mach 2.5, RT functionality implemented using "guest" OS	Mach 3.0 w/soft-real-time; RT IPC; instrumentation	Mach w/ISIS; Posix.4; "smart" distributed instrumentation	RT-Mach w/ISIS; Posix.5; B1 Trust	Fault-tolerant, RT Mach w/ISIS (e.g., over Galactica)
semi-automatic µCode compiler retargeting	mostly-automatic µCode compiler retargeting for specific processor types	µCode compilation with ~1.5-1:1 compactness ratio for special SP chips	fully automatic retargeting of µCode compiler with ~1:1 compactness	
* "traditional" application development	synthesis of algorithm impl.; domain-specific reuse	partial synthesis of application modules; auto-configuration of kernel	algorithm/µCode tradeoff tools	semi-automatic generation of µCode from algorithm sets

• This timeline assumes formalization of RASSP architecture
RT = Real Time
IPC = Inter-Process Communication
OS = Operating System

3.5 Information Infrastructure and Data Base

The recommended approach to the information infrastructure for RASSP is a federated network, as shown in Figure 3.5-1. The four basic areas; engineering, manufacturing, management, and logistics are capable of easy interchange and sharing of data and process services, reuse, traceability, configuration management, change notification, and sharing of cross-functional tools. Each basic area could be supported by a different framework.

The most important recommendations pertain to data models and tracking development of the F-22 Integrated Weapon System Data Base. The data models for signal processing need to be standardized, which requires time and investment. The RASSP program should develop the signal processing application protocol that is consistent with CFI and PDES. As the off-shot of this activity, the CFI and PDES standardization efforts should be focused and accelerated in the direction of RASSP. As stated above, the IWSDB program is synergistic with RASSP. However, it is important to track the IWSDB progress and, if necessary, direct and/or modify some if its parts for RASSP. For example, an important element of RASSP information infrastructure is the framework(s) to be used for each data base category. The engineering information flow could be handled by Mentor Falcon framework. The manufacturing data base might be best served by TeamNet. The maturity of these and any new products has to be monitored by RASSP. An important source of information about the maturity and applicability of data base software are the efforts by industry to incorporate them in the engineering/ manufacturing environment. For example, the TeamNet is installed on the computer network of Honeywell Commercial Flight Systems/Minneapolis Operation (CFS/MO), as shown in Figure 3.5-2.

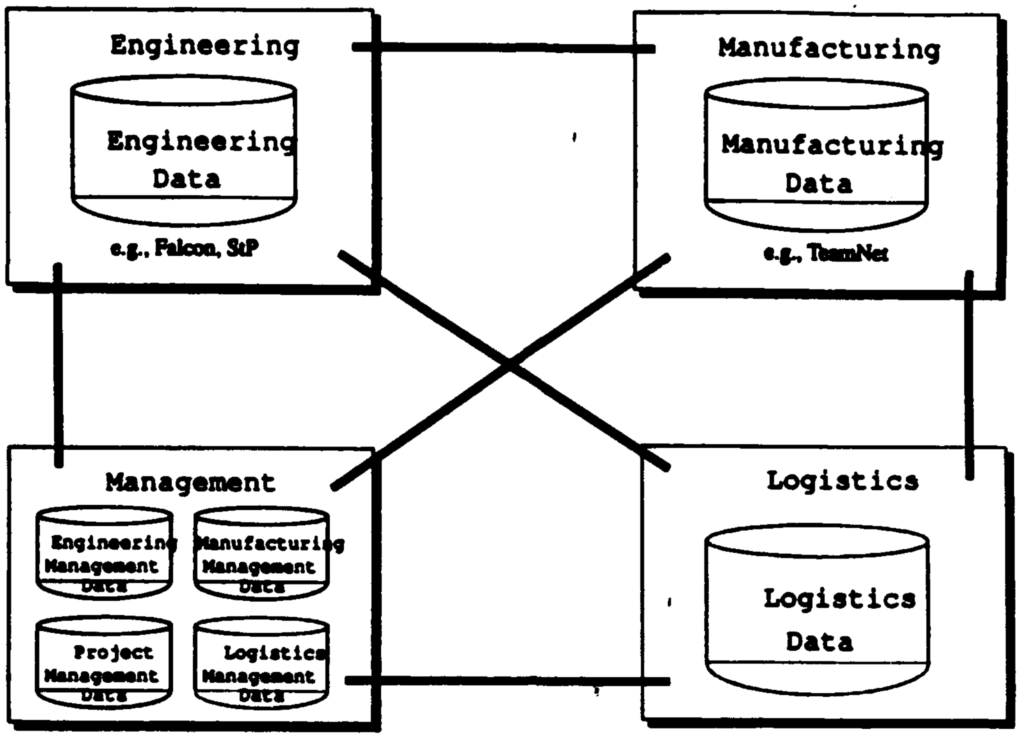


Figure 3.5-1. Federated Info. Infrastructure Strategy

The lesson learned so far at CFS/MO, Honeywell includes:

- Computer platform and network hardware needs to be assembled with integration in mind
- The optical memory media is adequate if supported by high performance server
- TeamNet handles the engineering change well (design-manufacturing interface)
- Beware of high hardware, software, and support labor cost

The other areas that require tracking of technology and product development are:

- Data services that suffer from immature distributed system technology, lack of object-oriented standard and poor performance

Over ride area network:

- Process services which require parallel domain specific standardization efforts
- Traceability, configuration management, change notification which are discontinuous between discipline and lacking in details (e.g.. change notification description)

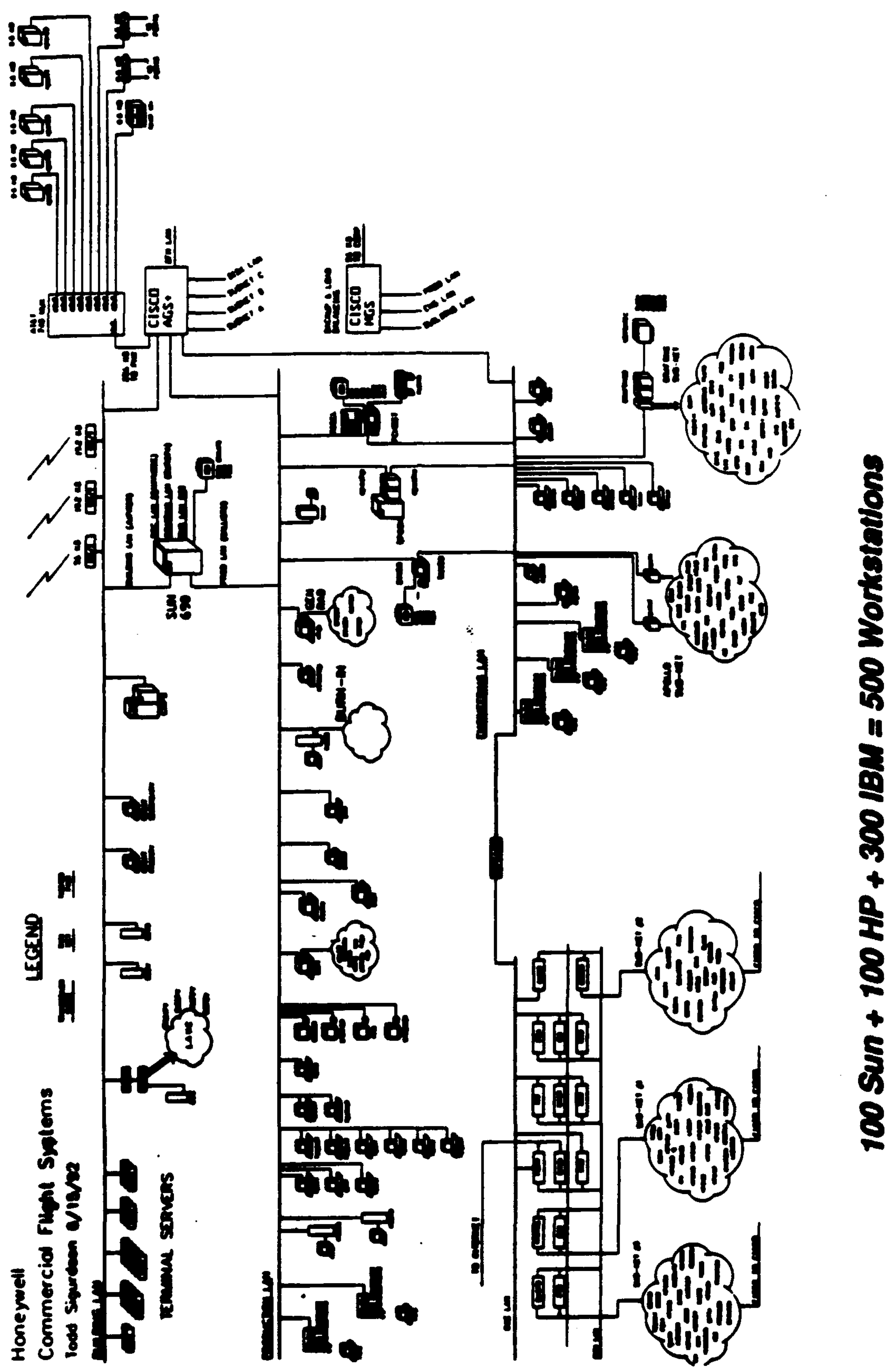


Figure 3.5-2. Computer Network at Honeywell Commercial Flight Systems Operation

3.6 Testing and Evaluation

Testing technology has recently undergone a revolution due to the sky-rocketing complexity of systems, modules, and components. The increasing complexity manifests itself in the increased number of components, increased number and types of interfaces, varied technologies, mixed digital/analog designs, sophisticated functionality, etc. On the other hand, the affordability drive of RASSP requires test simplicity/uniformity, high fault-tolerance, and a large presence of self-test and performance monitoring.

The testing technology has been discussed in detail in Subsection 2.6. The state-of-the art, programs, standards, issues, and recommendations in the areas of design, test equipment, and test technology are presented in Tables 2.6-1, 2.6-2, and 2.6-3, respectively.

The most important problems for RASSP testing are, test requirement and specifications, test program generation, and testability analysis. The testability analysis is limited on both system and chip levels to topology and structure information only. The derived approach is to include testability in the detailed trade-off analysis at the system level for both hardware and software. The design for testability and accessibility should be established and followed. A recommended approach is a comprehensive testability analysis with multiple-domain information models at higher levels and signal processing testability guidelines at more detailed levels. The test program generation is the second area requiring substantial RASSP participation. In spite of the ongoing programs (VTEST, TISS, ABET, and SS/REC) and standards (WAVES), the capabilities for tester independent capabilities and rapid development through reuse do not exist. The RASSP goal would be to generate the signal processing domain specific modeling approach and develop a model library leveraged by the ongoing programs. The third area in need of significant RASSP support is the test requirements and specifications. The issue is that the test requirement cannot be extracted or traced back to requirements in an efficient and consistent way. The solution is to develop an integrated test system environment, use test requirement models to support analysis, and implement domain-specific requirement reuse.

The remaining testing recommendations can be divided into supporting standards, encouraging vendor cooperation, and supporting research activities. The new standards are required for design and test beyond the IEEE 1149.1 into the analog, military, full-scan and other domains. The test equipment needs to be standardized especially in the area of MCM, HDI, and system testing. The standards and the vendor cooperation is needed in areas such as test simulation (especially analog and mixed digital/analog), ASIC and board test equipment, software test /verification tools, and "known good die" determination for MCM and HDI packaging.

The high performance platforms, hardware accelerators, and/or distributed simulations are required for test pattern generation and grading, test equipment at all levels (ASIC, MCM, board and system), and test simulation.

Finally, the existing test synthesis tools are inflexible and rigid. Thus, performance trade offs are not possible. RASSP should support the research in the area (partially in progress) at universities, MCC, and industry.

The roadmap of the test technology development in the three most critical areas, test requirement specification, test program generation, and testability analysis is shown in Table 3.6–1. The fast progression in these areas is dependent on RASSP funding. Table 3.6-1 represents the milestones of recommended program (see Subsection 2.6 for detailed discussion).

Table 3.6-1. RASSP Test Technology Progression

	92	93	94	95	96
Test requirements specifications	Manual MIL-Std 1519	Requirements and change tracking from system requirements through implementation code (EIP)	Formal models and language Requirements level reuse	Can extract from product models (TRSL, VTEST)	System level simulation of test requirements (SS/REC)
Test program generation	Some automatic capability, mostly manual, few standards—WAVES	ABET architecture target support	Architecture specific test requires driven generation (reuse) Topology driven diagnostic strategies	Automatic AI-based diagnostic strategies and generation	Multi-target auto generation from requirements and product models (vtest, trsl)
Testability analysis	Dependency-based model STAT, WSTA		Standard functional dependent model (AI-ESTATE)		Multi-domain (include ATE target) test Analysis

3.7 Manufacturing, Design/Manufacturing Interface, and Cost

Our study results have stressed the point that manufacturing must be included as a co-participant in the overall concurrent product development environment for RASSP. We see advanced, automated manufacturing technology playing a key role in the RASSP implementation effort. It will be a risk reducer through fault avoidance in the assembly of fine-pitch advanced technology boards; and it will be a cost reducer through its quicker assembly time and implementation of best commercial cost-avoidance practices such as JIT stocking, use of group technologies/parts, and activity based costing. This manufacturing focus is mainly concerned with printed circuit board assembly because we have assumed that ASICs, ASEMs/MCMs, and even the fabrication of the multi-layer printed circuit boards themselves, will be viewed as vendor supplied parts, these parts acquired through a vendor-networked infrastructure.

Many of the views expressed in this report have been distilled from visits to and observations of Honeywell product divisions and corporate planning activities. The accompanying videotape from Honeywell's Home and Building Control Division is but one example. However, our literature search and general awareness of industry trends all echo common themes concerning the directions that modern integrated enterprises are moving and the forces propelling them into the next century. The importance of process is a commonly heard theme.

3.7.1 The RASSP Manufacturing Process and Interfaces

The overall integrated product development process in general, and the manufacturing process in particular that has been described for RASSP implementation, are not "business as usual." Manufacturing is a new, flexible organization characterized by:

- Highly automated, planned for high volume assembly, yet amenable to a lot size of one
- Tightly coupling to design so single prototype units can be introduced and built "on the fly"
- Accommodation of change by employing agile, flexible manufacturing techniques
- Organization committed to concurrent engineering/integrated product development "best practices" and to ongoing process improvement.

Manufacturing must be a full participant in the RASSP cross-functional process and must participate throughout. Producibility considerations must be part of this process from beginning to end.

Within Honeywell we have observed a broad spectrum of manufacturing capabilities, from minimally automated production typical of some of our military-oriented divisions to totally automated "world class (commercial) production" illustrated by HBCD in the accompanying video. Figure 3.7-1 summarizes some of these capabilities and the SOA parts complexities anticipated in the not too distant future for commercial manufacturing. Figures 2.2-8 and 3.2-2 also provide glimpses of the packaging advances anticipated for military products like the Touchstone processor. To rapidly develop prototypes and products of these anticipated complexities, the traditional "toss it over the wall to manufacturing" will not work.

The design-manufacturing interface must be a real-time, bi-directional, networked interface with CAD, CAM, and CAT tools linked and with common access to an IPD data base. This type of with network and data base should be a part of the RASSP implementation phase.

Recommendations for standards to implement this environment are:

- Choose CAD, CAM, and CAT tools available on multiple platforms
- At a minimum, provide facilities for binary file transportability
- To facilitate file sharing between applications, encourage the use on non-proprietary file types such as DOS ASCII or postscript.
- Use a combination of NFS, Ethernet, and TCP-IP for wide area networking.

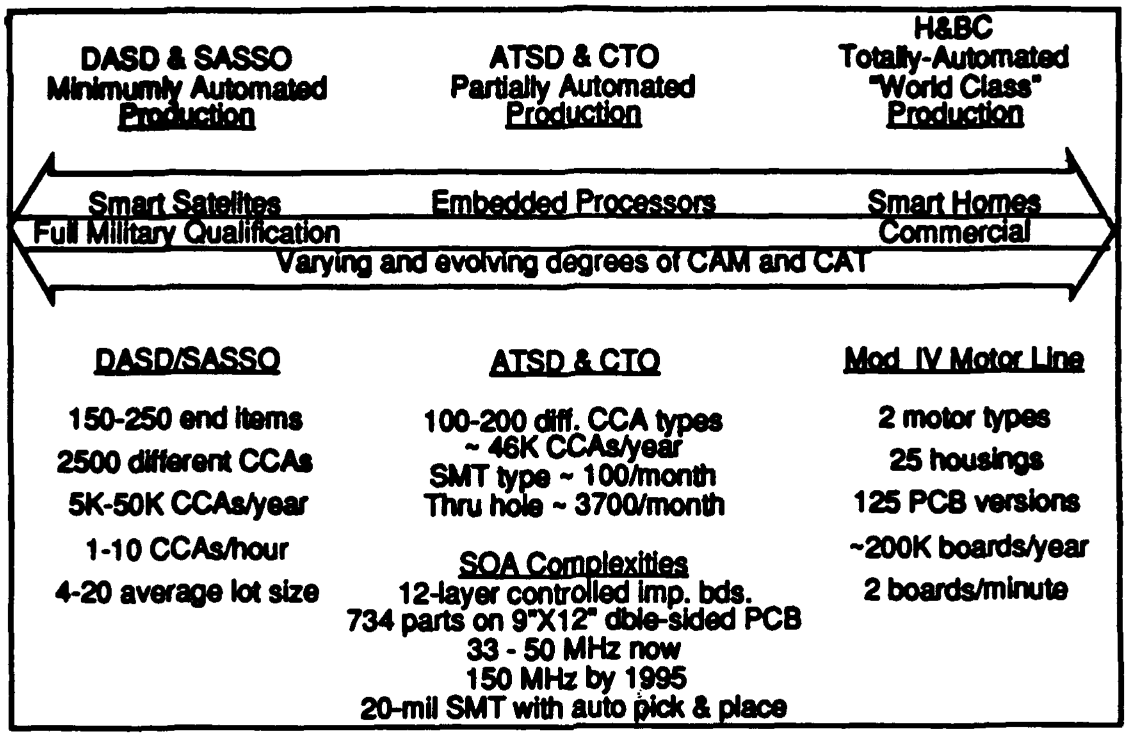


Figure 3.7-1 Spectrum of Manufacturing within Honeywell

3.7.2 Other Manufacturing Practices and Interfaces

We have recommended using "best commercial practices" as much as possible in the RASSP implementation phase. These practices include:

- Just-in-time parts procurement
- On-line parts storage
- Certified inventory storage through qualified suppliers
- Eliminate incoming parts inspection
- Buy parts from qualified manufacturers
- Perform qualification and reliability testing mainly at the box or assembly level
- Use statistical process control and continuous process improvement methods
- Use activity based costing techniques
- Ensure that field support is linked into and is part of this hole process
- Establish metrics so that cost and performance can be continuously monitored

A preference for using a commercial production facility for the RASSP implementation phase has been stated. If this cannot be accomplished, then we would recommend setting up a separate advanced assembly line for RASSP, apart from but in parallel with a similar commercial line. The synergism between these two lines would still be beneficial to both and a risk reducer to the RASSP program. Our long-range hope is that the barriers between military and commercial businesses can be overcome or eliminated.

Section 4
Significant Hardware and Software Concept Development

The development of new hardware and software was outside the RASSP Study scope. However, in the process of collecting and organizing the RASSP relevant information, we defined several new hardware and software design related concepts that might help in defining the RASSP process. In particular, the proposed software development and engineering process formalism might become a framework to integrate separate design tools and methodologies into one RASSP process.

4.1 Software Development and Production for Reuse

The RASSP program is driven by an affordability thrust. Therefore, the cost, efficiency, and turn-around time of software development need to be revisited. The need is further strengthened since the software constitutes more than fifty percent of the cost and development time of an advanced signal processor.

The software development for RASSP should evolve towards a reuse engineering process (REP) as defined below. The software product engineering process will be then adjusted to account for the new information flow as dictated by the REP. The flow of the software development process is shown in Figure 4.1-1.

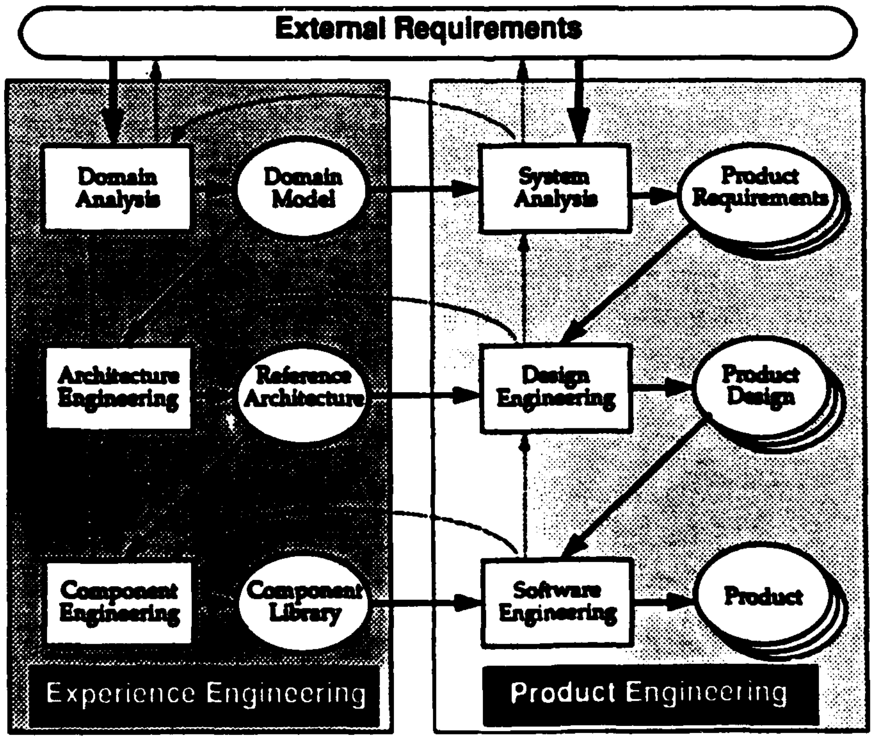


Figure 4.1-1. RASSP Software Development Process

Signal processing software development productivity and quality depends upon several technical factors (in addition to managerial factors):

- Refined methodology—Each design step must clearly add value to the product.
- Complexity management—Routine decisions must be automated out of the engineering perview.
- Design rationale capture—The reasons for decisions can sometimes be more important than the decisions themselves, especially if rework or quick iteration is required.
- Robust testing—Confidence in the proper operation of the software—degraded by human decision throughout the development process—must be restored through both structural and requirements-based tests.

While software offers greater flexibility and fewer intrinsic constraints than other engineering disciplines, software engineering is relatively immature. As a result, software systems often lack the established standards and conventions that guide systems architecture and design in other engineering disciplines, limiting our ability to produce software systems primarily by our ability to manage the complexities of very large-scale designs. That is, our software capability is determined almost entirely by our capability to support the software process.

The software development process has evolved towards progressively higher levels of abstraction in both programming language and design formalism. Recently, significant attention has been paid to developing methods for capturing and exploiting meta-knowledge, that is, knowledge about the software development process and its artifacts. Gathering the meta-knowledge—called domain engineering or reuse engineering—is on its way towards quickly becoming a science with specialized tools to support it.

The traditional software development process begins with system analysis which produces software requirements. The software requirements serve as the basis for developing the top-level architecture and design. Coders then take it the rest of the way into source code modules and finally into a product. In the more enlightened implementations, there will be feedback loops from any "downstream" process to one "upstream," even back to systems analysis. These feedback loops can, in principle, be quite tight, supporting quick, system-level "what ifs," or prototyping. However, without a disciplined approach to capturing all of the fruits of one's labor (e.g., generalizations, lessons learned, new approaches), each succeeding but similar software project is doomed to make insignificant improvements (at best) in quality and productivity.

Techniques are available and are now being applied to capture domain knowledge (expertise). While similar in approach to the now popular expert system, the aim of these techniques is not to emulate human advisors, but to continually accelerate the (software) product development process. Reuse engineering can be divided into three parts: domain analysis, architecture engineering and component engineering.

4.1.1 Reuse Engineering Process (REP)

The reuse engineering process defines the common requirements, architecture, and components that will guide and constrain the development of a family of related products. Although the focus within this process is on the family of products, as opposed to an individual product, the process does account for the necessary flows of information between the reuse and product engineering processes. These information flows are the primary means by which the products of the reuse engineering process are adapted to meet evolving requirements.

The REP is divided into three subprocesses: *Domain Analysis, Architecture Design,* and *Component Engineering*. Domain analysis produces a domain model for a family of systems (i.e., a description of *what* these systems must do). Architecture design produces a reference architecture for a family of systems (i.e., a description of *how* these systems are structured). Component engineering produces a set of reusable artifacts which may be combined according to the rules and conventions specified by the reference architecture to produce a working system. We describe each of these subprocesses in the following sections.

Domain Analysis—The domain model embodies a set reusable requirements, and represents a systematic classification of the principles and abstractions of a given domain, recorded using formal and semi-formal notations. One must carefully bound the domain of interest in order to keep the domain analysis task tractable. Domain models can usually be decomposed into a lattice of subdomains.

A domain model generalizes all of the systems in a particular application area in a way that transcends specific applications. Whereas systems analysis is concerned with the characteristics of a specific system, domain analysis is concerned with the characteristics of a family of similar systems.

In general, domain analysis is not a discovery process where new theories are developed and tested. Rather it is a process of codifying previous experiences. Domain analysis is also an evolutionary process in which a domain model is incrementally refined over time based on feedback from domain experts and users of the reusable artifacts derived from the model.

Architecture Design—Given a domain model that describes the requirements for a particular problem domain, architecture design is the process of partitioning those requirements and allocating them to elements of a reference architecture. The reference architecture is a reusable design which defines how a family of systems is structured (i.e., the application and execution contexts). That is, one can view the reference architecture as the outline of a partially defined system structure in which features common to all members of the system family have been factored out and established as fixed characteristics of the structure and differences between family members have been parameterized to allow for controlled customization. Note that a reference architecture may be hierarchically defined along traditional system/subsystem lines. At each level of abstraction, one would see requirements and architecture models (i.e., one person's requirement is another person's design). Done properly, such decomposition facilitates testing and system integration.

The context embodied in the reference architecture consists of both the application context and the execution context. The application context encompasses all the application-related design decisions for an organization's products within a given problem domain. The execution context encompasses all the environmental factors that impact the design and operation of a software system. By establishing and documenting this context, a reference architecture enables standardization of internal and external interfaces, and methods and procedures of operation, which in turn facilitates the development of components which can be reused across a family of systems which conform to the specified standards.

D.A. Perry proposes a model of software architectures in which an architecture consists of elements, form, and rationale [PERRY91]. In this model, there are three classes of architectural *elements*: data elements, processing elements, and connecting elements. Data elements contain the information that flows through a system. Processing elements perform transformations on data elements. Connecting elements provide communication channels through which processing elements exchange data elements. Examples of connecting elements include procedure call, shared memory, and message passing. Architectural *form* consists of weighted properties and relationships. Properties place constraints on architectural elements. For example, an architecture may specify a processing element that is to sort a collection of values. Properties might be used to specify time and space constraints on that element. Later, during system construction, only components meeting those constraints could be bound to that element. Relationships constrain the placement of architectural elements by constraining how the different elements may interact and how they are organized with respect to each other in an architecture. For example, a relationship might specify a communication channel with special timing and reliability constraints between two elements. Weighting indicates either the importance of a property or relationship or the need to select among alternatives (some may be preferred over others). Finally, the architectural *rationale* captures the motivation underlying the design decisions embodied in an architecture. That is, the rationale explicates why an architecture is structured one way as opposed to another (note that this rationale necessarily ties an architecture back to the requirements defined by the associated domain model).

Such an architecture supports both multiple views and analysis. With respect to views, a process view emphasizes the data flow through the processing elements (as well as some aspects of the intermediate connecting elements). A data view emphasizes the processing flow (with less emphasis on connecting elements). Other views can be envisioned, such as timing or reliability, which are derived from appropriate properties and relationships. Different views provide varying perspectives on the architecture that leads to a better understanding of the process. Differing views may also help uncover different types of deficiencies. With respect to analysis, depending on the domain-specific properties and relationships that annotate an architecture and their formality, one may analyze an architecture in various ways prior to full system construction. For example, consider an architecture for a system of preemptive, fixed priority tasks (e.g., a flight management system). Given appropriate interconnection, timing, and hardware/software binding information, scheduling analysis can be used to determine if the specified schedule can be met. Such analysis allows developers to perform "what-if" trade-off analyses at a systems level to ensure that the system-level design is adequate before the system is built.

Note that the RASSP software design process effectively merges the traditional top-down development approach with a bottom-up perspective. That is, basing a system on a predetermined architecture emphasizes *analysis*, while composing the system from components emphasizes *synthesis*.

Component Engineering—Taken together, the domain model and the reference architecture define the context necessary to permit the definition and implementation of components that will be applicable to the construction of a variety of products within the given product family (i.e., reusable components). Each component is designed to fill a specific role within a product implementation, as defined by the reference architecture. For any given role, there may exist several reusable components, that are applicable in differing circumstances as determined by the properties and relationships associated with that role within the architecture.

A component may be viewed as an interface specification analogous to an [Ada83] package specification. In particular, a component defines a parameterized behavior. Parameterization allows controlled tailoring of the component. Each component would also be annotated with attributes similar to the properties and relationships used to annotate the reference architecture thereby supporting a pattern matching capability for determining if a given component will satisfy a given architectural role.

Each component also has several associated artifacts. In particular, a component will typically have one or more implementations. A component implementation may consist of a single subprogram, a set of executable processes, or anything in between, as long as it properly fills its intended role within the reference architecture. Most reusable components, however, will have implementations that are analogous to an Ada package body. In some cases, component implementations may be automatically generated from component specifications. Note that an implementation will also be annotated with attributes similar to the properties and relationships used to annotate the reference architecture. A component's documentation will consist, at a minimum, a description of the component's purpose (what role(s) within the reference architecture it fulfills), a summary description of its principles of operation, its derivation/ modification history, and its implementation usage history (what products within the product family have used various implementations of this component).[1] Each component will have an associated set of tests. These tests should be sufficient to exercise all of the defined functions of the component (black box testing); the tests should exercise the component for proper functioning in the presence of valid inputs (positive testing), and also for acceptable behavior in the presence of invalid or missing inputs (negative testing); for some components, it may also be necessary to verify that every possible unique path of execution through a component implementation has been tested (white box testing).

Components and their associated artifacts are maintained in the component library. This library is subject to strict change, version, configuration, and quality control. This last point is

[1]Some of this documentation, e.g., the derivation/modification history and the usage history, may be automatically collected and maintained by the software engineering CASE environment.

especially important—to be maximally effective, the component library contents must be of unimpeachable quality. Software engineers must have almost absolute faith in the quality of the artifacts contained, otherwise, these assets will not be used. Therefore, it is imperative that all components, implementations, documentation, and tests be subjected to rigorous quality control *before* they are entered into the component library, and they must be re-certified whenever they are modified.

Ideas for new components and for changes to existing components will undoubtedly be generated during the implementation of various products. These ideas should be fed back into the component engineering process, where they can be properly evaluated in light of the reference architecture. This may result in the construction of new components, modifications to existing components, or suggestions for changes to the reference architecture (which are fed back to the architecture design process).

4.1.2 Product Engineering Process

The product engineering process (PEP) defines the means by which a single product within the product family is produced. Although the focus within this process is on the delivered product, as opposed to the family, the process does account for the necessary flows of information between the product and reuse engineering processes.

The PEP is divided into three subprocesses; *System Analysis, Product Design,* and *Software Engineering*. Although these names are similar to the names of the subprocesses of the traditional product development process, the specifics of the activities to be performed are somewhat different in the presence of a reuse engineering process (REP).

System Analysis—The traditional software development process usually begins with system analysis. According to [COAD91]

> "Analysis is the study of a [problem], leading to a specification of externally observable behavior; a complete, consistent, and feasible statement of what is needed; a coverage of both functional and quantified operational characteristics (e.g., reliability, availability, performance)."

In the context of a REP, the system analysis process is constrained and guided by the existence of a pre-determined domain model, which provides a standard, parameterized, but incomplete, framework upon which the requirements of the new product are based. The PEP system analysis process consists of developing a product requirements specification for the new product, in accordance with the pre-determined requirements and constraints contained in the domain model, that meets the needs of the specific problem at hand.

The domain model contains the "pre-computed" analysis results which are common to *all* problems within the given domain. It captures the results of an analysis of a problem domain, as opposed to an analysis of a single problem within the domain. The domain model provides a structure for subsequent (problem-specific) analyses, and it constrains the choices made during those analyses by imposing various pre-determined results. These pre-determined results, in effect, determine the basic parameters of all products (in the given problem domain) that are constructed with reference to the given domain model.

It is unwise for the system analysis effort to simply start with the domain model and then modify it arbitrarily to produce the product requirements specification for a new product, for three reasons:

1. It will be difficult or impossible for the knowledge gained as a result of the new analysis effort to be assimilated into the domain model for use by subsequent products;

2. It may become very difficult or impossible to incorporate existing components from the component library into the product architecture resulting from the new requirements specification, since the contextual guarantees provided by the reference architecture derived from the domain model may be violated; and,

3. The introduction of a new product into the product family—not based on a common domain model and its associated reference architecture—will lead to increased maintenance costs due to the extra effort required by the organization to maintain and support a variety of products that are related, but that are sufficiently dissimilar to prevent the easy transfer of knowledge, techniques, and personnel across product boundaries.

Instead, the system analysis effort must work within the confines of the existing domain model, whenever possible. When due to rapidly changing customer, market, or regulatory demands, a new project is faced with the task of incorporating new characteristics that are outside the bounds of the existing domain model, the organization *must* constrain the development process sufficiently to ensure that these new requirements are fed into the domain analysis process and the domain model is appropriately updated. Then, and only then, can the system analysis for an individual product proceed.

The costs of the domain analysis process are amortized over the family of products that utilize the results (i.e., the domain model), which will decrease the cost and increase the quality of the results of the system analysis process. The domain model reduces the necessary system analysis effort for any single product by capturing the characteristics common to all problem solutions in the domain in a single analysis effort, and reusing these results in many subsequent problem-specific analyses.

Each new product benefits from the organization's experience and product improvements obtained from previous analysis efforts in the given problem domain. In any given system analysis effort, new requirements or characteristics may be discovered that are not represented in the existing problem domain model. This information is fed back to the domain analysis process, where it is analyzed and incorporated into the domain model in a consistent manner. Thus, as new problem characteristics—and their approved solutions—are discovered, they are incorporated into the organization's formal software development process for the benefit of the current and all subsequent products.

Product Design and Software Engineering—The product design and software engineering processes take on a slightly different character in the context of the REP. The reference architecture expresses a "pre-computed" engineering design common to *all* problems within the given domain.

The software engineering process consists of four steps:

1. Understanding the previously developed product architecture, reference architecture, and library components;

2. Selecting an appropriate set of components from the library to fulfill specific roles defined by the reference architecture;

3. Composing the selected components according to the approaches and constraints imposed by the product architecture; and

4. Integrating the selected components, underlying operating system software, and the newly developed code[2] into a functional, documented, and tested product that conforms to the given product specifications.

The component library provides a set of pre-developed components that are applicable to building software products in the given domain. These components provide a standard set of solutions and conform to the design choices and constraints imposed by the reference architecture. These components become the basic "reusable components." Their reusability is guaranteed solely by their conformance to the conventions and constraints imposed by the domain model and the reference architecture.

The software engineering effort must work within the confines of the existing reference architecture and available library components, whenever possible. When, for whatever reason, a new project is faced with the task of incorporating new product characteristics that are outside the bounds of the existing reference architecture and/or the available selection of reusable components, the organization should constrain the development process sufficiently to ensure that these needs are fed into the component engineering process. This will facilitate the development of high quality, reusable components that are certified to conform to the reference architecture and tht are applicable to the entire product family.

---

[2]At first, *some* new software will be written to "bridge the gaps" between the reused components and/or to meet unique needs for each new product. However, as the domain model, reference architecture, and library of reusable components improve over time, the need for such "glue code" will diminish. It will (probably) never vanish, but will instead become an increasingly minor portion of the overall product software. Thus, the production of new code becomes a relatively minor aspect of the job of the "product software engineer."

4.2 Design Process Formalism

The design process formalism described within is not new but is fundamental to the engineering problem solving approach. The three basic steps that make up this process, *definition*, *implementation*, and *verification*, are repeated over and over again throughout the design process as the process progresses and the design problem is hierarchically decomposed.

- Definition—Given a set of requirements and associated data, define the design/problem and its parameters.
- Implementation—Develop a candidate design model or problem solution, by optimizing iteration as required. This implementation step by its nature introduces the concept of hierarchy, the idea of an element *composed* of *sub-elements*.
- Verification—Verification has two important aspects. One aspect is verifying the consistency of the design/problem representations as the design is hierarchically decomposed and/or integrated throughout the design process. The other aspect of verification is checking that an actual result from a step in the design process realizes the intended result. In both cases, one iterates the process until a design/solution is completed and it is consistent throughout.

This design process formalism is diagrammed in Figure 4.2-1; namely, the concept of an element with inputs and outputs that can be decomposed into similar "child" elements also with inputs and outputs.

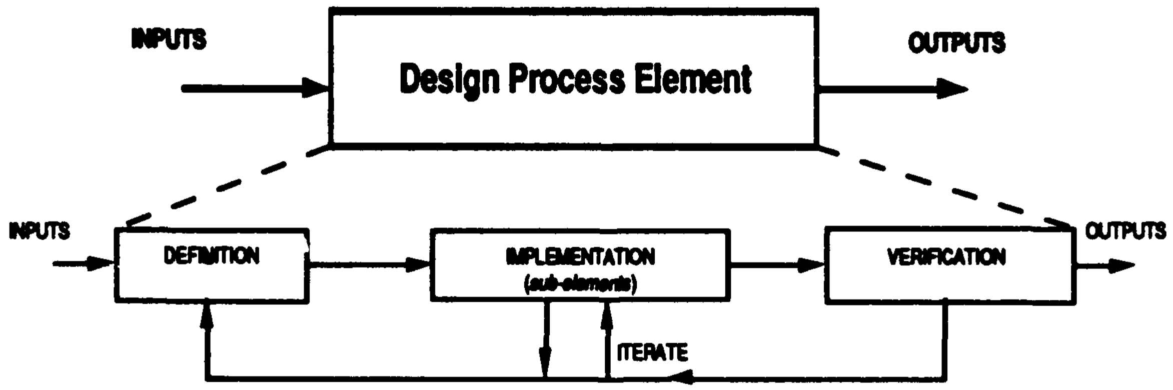


Figure 4.2-1. A Design Process Element and Its Decomposition

The design itself, both software or hardware, must be described in a canonical form that can provide a measure of the design's completeness. The design attributes of *form*, *fit*, and *function* (F^3) will be used as the specifications necessary to completely define a design. Again this is nothing new. The airline industry has been using F^3 practices for years, and the DoD has occasionally applied it to new systems.

- Form—The design's physical description; i.e., its makeup and composition, size, weight, power, material, packaging, lines of code, bits per word, etc.;

- Fit—The design's interfaces to the outside world; i.e., its architecture or structural interfaces, buses, I/Os, connectors, standard interface formats;

- Function—The description of the design's behavior; i.e., what it does, timing, test, operating modes, software functionality, etc.

At any and all levels of a hierarchical design, each element must have form, fit, and functional descriptions for that element, whether hardware or software, to be completely defined.

When viewed in the larger context of concurrent product development, the attributes of form, fit, and function provide a concise basis for introducing the "ilities" into the design process right up front in the requirements capture and performance capture phases of the process. As the design is subsequently defined and hierarchically decomposed, consideration of the "ilities" on the design can be imposed and consistently traced throughout the design and configuration management processes. Design decisions thus include consideration of manufacturability, quality, testability, reliability, affordability, etc., as an integral part of the design process. Again this can apply to both hardware and software.

References

[Ada83] United States Department of Defense, Reference Manual for the Ada Programming Language, ANSI/MIL-STD-1815A, 1983.

[COAD91] Coad, P., E. Yourdon, *Object-Oriented Analysis*, 2nd Ed., Yourdon Press, Englewood Cliffs, NJ 07632, 1991.

[PERRY91] Perry, D., A. Wolf, "Software Architecture," submitted for publication, 1991.

Section 5
Implications for Further Research

The further research and development directions are amply described in Sections 1 and 3 with technical details discussed in Section 2 and 4. These recommended research and development tasks need to be performed by the RASSP programs in parallel. Therefore, at this time there are no additional topics identified for further research.

Appendix A
Compendium of Related Work

The compendium has been divided in the sections corresponding to subsections of Chapter 2 of the final report. The publications with asterisk originated at Honeywell.

1. Target System Selection

ATR System:

Twelfth Annual IEEE/AESS Dayton Chapter Symposium: Future Avionics .."What Direction Will It Take?" IEEE; 4 December 1991.

A. Wilson, "A New Image for Image Processing," Tulsa, OK, Military & Aerospace Electronics, 17 August 1992. *Note: Cover Story. Learning from Desert Storm, the U.S. military is upgrading its image processing capabilities.*

H. Lapp, "ATR System: An Air Force Perspective," SPIE, Vol. 750, Infrared System and Components, 1987. *This paper discussed the second generation FLIR sensor and the system impacts on automatic target recognizers.*

F. Shields, "Automatic Target Recognizer a Center for Night Vision and Electro Optics Perspective," SPIE, Vol. 750, Infrared System and Components, 1987. *This paper addressed the importance of a unified percepts on human operator, modified tactics, multi-sensor algorithms for a successful ATR.*

J. Knecht, "ATR A Navy Perspective," SPIE Vol. 750, Infrared System and Components, 1987. *This paper discussed three missions that an ATR found useful in Navy.*

H. Robert, "Practical Issues In The Multisensor Target Recognition," SPIE Proceedings, Vol. 1306, 1990. *This paper assessed several trends and conditions in multisensor target recognition and recommended actions to some of the concerns.*

W. Delashmit, "Present Status and Future Needs for Automatic Target Recognizers," Proc. SPIE, Vol. 1075, Digital Image Processing Applications, 1989. *This paper addressed the status of the ATRs and indicated areas which required new techniques to improve system performance.*

W. Brown, C. Swonger, "A Prospectus for Automatic Target Recognition," IEEE Trans. Aerospace and Electronic Systems, Vol. 25, No. 3, May 1989. *This paper discussed various aspects of an ATR system, and supported a favorable assessment of the power and importance of the ATR technology.*

*J. Hodapp, "Processor Miniaturization to Support Real-Time Target Acquisition," 24th Asilomar Conference on Signals, Systems and Computers, pp. 579-82, Vol. 2, 1990. *This paper described two architectures that realized miniaturized processors for use in real time image and signal processing.*

C. Roark, A. Harper, "Aladdin Software Support," Proc. NAECON, IEEE, 1991. *This paper described the software development toolset and run-time environment to support the application development in the Aladdin processor.*

*S. Lidke, et al., "Multi-Function Target Acquisition Processor (MTAP) Final Report," CNVEO, 1989. *This is the final report on the MTAP program. The report described the algorithms and hardware used to build a real-time ATR system.*

*S. Savitt, et, al., "Image Sensor Autoprocessor," Final Report, Vol. 3, prepared for Air Force Avionics Lab, Wright Patterson Air Force Base, July 1989. *This report described the algorithms and proposed hardware to build an ATR system.*

__ATR Algorithms__ (See IEEE Transactions on Pattern Analysis for more):

J. Verly, et al., "A Model-Based System for Automatic Target Recognition," Proc. SPIE, Vol. 1471, Automatic Object Recognition, 1991. *This paper described a 3D model-based target matching technique in recognizing targets in a laser radar imagery.*

*H. Nasr, H. Amehdi, "Model-Based Automatic Target Recognition Development Tools," Proc. SPIE, Automatic Objective Recognition, 1991. *This paper described a toolset for developing a model-based iconically reconfigurable object recognition system.*

G. Clark, et al., "Gabor Transforms and Neural Networks for Automatic Target Recognition," Proc. Workshop on Neural Networks, February 1991. *This paper described Gabor filter in performing robust recognition which might subject to rotation, scale, and translation.*

*F. Sadjadi, H. Nasr, "A Technique for Automatic Design of Image Segmentation Algorithms," Proc. SPIE, Aerospace Pattern Recognition, Vol. 1098, 1989. *This paper discussed a system concept for automatic design of segmentation algorithms based on metrics and image primitives.*

T. Theis, A. Akerman, "Comparison of Model Based, Vision, Statistical Based and Neural Net Based ATRs," IEEE, NAECON, Vol. 4, 1989. *This paper compared the three approaches at a high system level, and found that the difference was in the representations of targets which affected the implementation and performance flexibility in meeting new threats.*

*S. Lidke, M. Haskett, "Lessons Learned From a Commercial Module Approach to Real-time ATR Development," Conference on Image Understanding in the '90s: Building Systems that Work, SPIE, Vol. 1406, 1991. *This paper described the advantages and disadvantages of building an ATR system solely from commercial VME hardware modules.*

L. Napolitano, et al., "A Special-purpose Computer for Automatic Target Recognition," Proc. Int'l Conf. on Acoustics, Speech and Signal Processing, Vol. 3, May 1989.

D. Casasent, "Optical Pattern Recognition and AI Algorithms and Architecture for ATR and Computer Vision," Proc. SPIE, Vol. 755, 1987.

__Test and Performance Evaluation of ATR Systems:__

G. Currie, "Standard Test Targets for Automatic Target Recognitions," Proc. SPIE, Vol. 1307, Electro-optical Materials for Switches, Coatings, Sensor Optics and Detectors, 1990. *This paper reviews methods intended to test ATR and databases needed to support these methods.*

The following documents were issued by the ATR Working Group with Honeywell participation regarding specifications, standards and definitions:

"ATR Definitions, Performance Measures and Image Metrics," ATRWG, 86-001.

"Digital Image Data Exchange Format Specification," ATRWG, 86-002.

"ATRWG Format Specification for 9 Track Computer Tapes," ATRWG, 86-003.

"High Bit Rate Recorder Test Tape Specification," ATRWG, 87-001

"ATRWG Artificial Intelligence Committee Lexicon," ATRWG, 87-003

"Data Collection Guidelines for ATR Development," ATRWG, 88-002

"FLIR Characterization Guidelines for Test Data Acquisition," ATRWG, 88-004.

Appendix I to Doc. No. 86-001," ATRWG, 88-005.

"Applications of confidence Intervals to ATR Performance Evaluation," ATRWG, 88-006.

"Requirements for Representative Data in ATR System Development and Evaluation," ATRWG, 90-001.

"Data Collection Guidelines for ATRWG Laser Version," ATRWG, 90-002.

"Image Metrics," ATRWG, 90-003.

"Manned Aircraft versus Relocatable Target Strawman Specification," ATRWG, 91-001.

2. System Development

G. Hays, D. Wineberg, "Commilitary-Another New World Order," IEEE AES Systems Magazine, p. 16, September 1992. *Part selection, environmental conditions, levels of inspection and test, configuration control of commercial parts and practices applied to military systems (modular avionics radar).*

D. Felt, A. Mones, T. Poulin, "MCM System Performance Analysis," *Trade-off tool for cost vs. technology optimization.*

J. Biancini, "Implementing Concurrent Engineering for Surface Mount Technology," Electronic Packaging & Production, August 1992. *Concurrent Engineering and Design for Manufacturability are Integral to a Successful SMT Manufacturing Operation.*

P. Verhofstadt, "Current Trends in Design Science Research, " Semiconductor Research Corporation Newsletter, Vol. 10, No. 9, September 1992. *Path to meet the following goals: design product with 10^8 device complexity, 18 month design cycle, hardware/software codesign.*

D. Maliniak, Future Packaging Depends Heavily on Materials," Electronic Design, p. 83, January 9, 1992. *Packaging issues of 500 MHz and 1000 I/O ICs and MCMs.*

L. Burgess, "PLDs and FPGAs Work Their Way Into Systems," Military and Aerospace Electronics, p. 46, March/April 1992. *The gate densities above 10,000 and I/O speed up to 200 MHz combined with low design cost bode well for PLD/FPGAs for glue logic.*

J. Biancini, "Concurrent Design for Manufacturability," SuperNova Corporation, Minnetonka, MN 55345-2191. *Note: Course description and itinerary.*

*"Design-for-testability Guidelines for Printed Wiring Assemblies," Honeywell, December 30, 1988. *Note: Version 0.1.*

J. East, "An Imperative for the '90s,Æ ASIC & EDA, July 1992." *Technology transparent design may be the key to simplifying the design process.*

S. Evanczuk, "Concurrent Engineering: Design's New Look," Manhasset, NY 11030, High Performance System, April 1990. *Note: Whole issue.*

S. Galatowitch, "DESC Means Mil-Spec, Mil-Std," Englewood, CO 80111: Defense Electronics, Vol. 24, No. 7, July 1992. *Electronic components that meet rigid military specifications are mandated for use in many systems, but by what process are they militarized, how is reliability guaranteed, and how do they end up in the hands of a system integrator?*

R. Goering, "Designers Reach For a Higher Level," Manhasset, NY: Electronic Engineering Times, 3 August 1992. *Note: part 1 of 2.*

R. Goering, "Emerging Tools Aid High-level Design," Manhasset, NY: Electronic engineering Times, 10 August 1992. *Note: part 2 of 2,.*

R. Goering, "Rapid Prototyping: How To Stay On Course," Manhasset, NY 11030, High Performance Systems, October 1989.

L. Gullman, "Rapid Product Design and Development: Tolls and Strategies for Shrinking Time to Market," SRI International, December 1991.

K. Hamel, "State Mate Sales Brochure," Burlington, MA 01803: i-Logix.

"Handbook of Concurrent Engineering," McDonnell Douglas Missile Systems Company, December 1990.

B. Henderson, "CAD/CAM Systems Transform Aerospace Engineering." Aviation Week and Space Technology, 22 June 1992.

V. Hunt, "Enterprise Integration Sourcebook: The Integration of CALS, CE, T M, PDES, RAMP, and CIM," San Diego, CA 92101: Academic Press, Inc., 1991.

*T. Jenne, J. Hunger, P. Zumsteg, A. Anderson, K. Mikkilineni," 1990 Manufacturing/ Engineering Architecture Project Final Report," Minneapolis, MN 55418: Honeywell, 30 January 1991. *Note: Volume 1: product design process model.*

*T. Jenne, J. Hunger, P. Zumsteg, A. Anderson, K. Mikkilineni," 1990 Manufacturing/ Engineering Architecture Project Final Report," Minneapolis, MN 55418: Honeywell, 30 January 1991. *Note: Volume 2: appendices to the product design process model.*

L. Litton, D. Hall, K. Hutchinson, D. Hoffman, S. Evanczuk, P. Sullivan, "First Principles of Concurrent Engineering: A Competitive Strategy for Electronic Product Development," Springfield, VA: CALS/Concurrent Engineering Task Group—Electronics Systems, 30 September 1991. *Note: Reproduced by U.S. Department of Commerce, National Technical Information Service. Draft Copy CALS Technical Report 005.*

N. McEachron, R. Aldrige Tara, "Reducing Time to Market: Selecting and Executing Accelerated Development Strategies," SRI International, March 1990.

L. Menker, "Results of the Aeronautical Systems Division Critical Process Team on Integrated Product Development," Wright-Patterson AFB, OH 45433-6503, November 1990.

J. Pennel, M. Slusarczuk, "An Annotated Reading List for Concurrent Engineering," Alexandria, VA, Institute for Defense Analyses, July 1989. *Note: IDA Document D-571. Documents acquired by Defense Technical Information Center (DTIC).*

Sales Brochure, "Interactive Development Environments."

Sales Brochure, "Vantage."

S. Schultz, "The HDL Wars: Who Loses?" ASIC & EDA, July 1992. *Verilog vs. VHDL is shaping up as a classic confrontation - established incumbent faces a highly touted challenger. Will the winner really take all?*

S. Schultz, "Finding the Best EDA Vendor," Los Altos, CA, ASIC & EDA, August 1992. *Reality vs. Illusion is more than just a concept you picked up in an English Lit class.*

SES/Workbench Sales Brochure, "Scientific and Engineering Software, Inc."

S. Shina, R. Redy, R. Wood, K. Cleetus, J. Turino, "Concurrent Engineering," New York, NY 10017, IEEE Spectrum, Vol. 28, No. 7, July 1991.

"SMT Conversion: One Company's Experience," Libertyville, IL, Surface Mount Technology, August 1992. *Note: Whole issue.*

System Design Station Sales Brochure, Mentor Graphics, 1992.

VHDL Self Start Kit Sales Brochure, Nashua, NJ 03060, Topdown Design Solutions, 1992.

G. Watson, "MIL Reliability: A New Approach," IEEE Spectrum, August 1992. *Traditionally faulted for not predicting reliability accurately, MIL-HDBK-217 now has an alternative waiting in the wings.*

R. Winner, J. Pennelee, H. Bertrand, M. Slusarczuk, "The Role of Concurrent Engineering in Weapons System Acquisition," Alexandria, VA: Institute for Defense Analyses, December 1988. *Note: IDA Report R-338.*

R. Winner, et al., "The Role of Concurrent Engineering in Weapons System Acquisition," IDA Report R-338, Institute for Defense Analyses, Alexandria, VA (available through NTIS), 1988.

R. Sprague, K. Singh, "Concurrent Engineering in Product Development," Design and Test of Computer, IEEE, March 1991.

C. Kelly, et al., "Findings of the U.S. Department of Defense Technology Assessment Team of Japanese Manufacturing Technology," Tech Report CSDL-R-2161, US DoD, Washington, DC, June 1989.

A. Rosenblatt, G. Watson, "Special Report on Current Engineering," Spectrum, IEEE, July 1991.

3. Hardware Development

P. Verhofstadt, "Current Trends in Design Science Research," Semiconductor Research Corporation Newsletter, Vol. 10, No. 9, September 1992. *Path to meet the following goals: design product with 10^8 device complexity, 18 month design cycle, hardware/software codesign.*

29th ACM/IEEE Design Automation Conference. Los Alamitos, CA 9020-1264: IEEE Computer Society Press; 1992; ISBN: 0-8186-2822-7 (case).

"Automated Circuit Design," Procedure and Quality Assurance Manual, 15 July 1991. *Note: There are two pieces involved - the manual and a sales brochure. Created to outline procedures/quality controls at ACD.*

P. Plansky, "Obstructions to ASIC Design," ASIC & EDA, July 1992. *Tools, test, and the human factor can all stand in the way of successful ASIC design.*

Sales Brochure: Synopsys.

Sales Brochure: Vantage.

S. Schultz, "The HDL Wars: Who Loses?" ASIC & EDA, July 1992. *Verilog vs. VHDL is shaping up as a classic confrontation - established incumbent faces a highly touted challenger. Will the winner really take all?*

S. Schultz, "Finding the Best EDA Vendor," Los Altos, CA, ASIC & EDA, August 1992. *Reality vs. Illusion is more than just a concept you picked up in an English Lit class.*

4. Software Development

Digital Equipment Corporation. Cohesion. *Advertisement for DEC Cohesion CASE tool.*

Integrated Design Automation System Product Description Summary Sales Brochure, JRS Research Laboratories Inc., 1036 W. Taft Ave., Orange, CA 92665, June 1988.

S. Pfleeger, "Measuring Software Reliability," IEEE Spectrum, August 1992. *Code must be reliable from the surprisingly divergent viewpoints of software developers, testers, and users.*

SES/Workbench Sales Brochure, "Scientific and Engineering Software, Inc." *Note: two copies.*

United States Department of Defense, Reference Manual for the Ada Programming Language, ANSI/MIL-STD-1815A, 1983.

W. Agresty, F. McGarry, The Minnowbrook Workshop on Software Reuse: A Summary Report, Contract NAS 5-31500, Computer Sciences Corporation, Systems Sciences Division, 4600 Powder Mill Rd., Beltsville, MD 20705, March 1988.

K. Birman, T. Joseph, "Exploiting Virtual Synchrony in Distributed Systems," Proc. of the 11th ACM Symposium on Operating Systems Principles, Austin, TX, pp. 123-138, November 1987.

B. Boehm, Software Engineering Economics, Prentice-Hall, Englewood Cliffs, NJ, 1981.

G. Booch, "Object-Oriented Development," IEEE Trans. Software Engineering, 12(2), pp. 211-221, February 1986.

F. Brooks, "The Mythical Man-Month," Addison-Wesley, Reading, MA, 1972.

P. Coad, E. Yourdon, "Object-Oriented Analysis," 2nd Ed., Yourdon Press, Englewood Cliffs, NJ 07632, 1991.

P. Spiby (ed), "EXPRESS Language Reference Manual," Document N14, ISO TC184/SC4/WG5, June 25, 1991.

M. Hammer, "The Last Word," Computerworld Premier 100, p. ~80, September 1991.

R. Holibaugh, "Joint Integrated Avionics Working Group (JIAWG) Domain Analysis Method," DRAFT, 1990.

W. Humphrey, "Characterizing the Software Process: A Maturity Framework," IEEE Software, 5(2), pp. 73-79, March 1988.

A. Jawarski, F. Hills, T. Durek, S. Faulk, J. Gaffney, "A Domain Analysis Process," Technical Report DOMAIN_ANALYSIS-90001-N, Software Productivity Consortium, SPC Building, 2214 Rock Hill Rd., Herndon, VA 22070, 1990.

JRS Research Laboratories, "Integrated Design Automation Systems (IDAS)," product description summary, 11 pages, June 1988.

J. Keller, "Why Unix Is Winning the Real-Time Race," Military & Aerospace Electronics, 9(2), pp. 26-28, September 1991.

T. King, "Software Reuse: Concepts and Issues," Technical Report CS-R92-007, Honeywell Systems & Research Center, 3660 Technology Drive, Minneapolis, MN 55418, April 1992.

J. Neighbors, "Software Construction Using Components," Ph.D. Thesis, Dept. of Info. and Comp. Sci., UC Irvine, 1981.

J. Neighbors, "The Draco Approach to Constructing Software from Reuseable Components," IEEE Trans. Software Engineering, 10(5), pp. 564-574, September 1984.

D. Perry, A. Wolf, "Software Architecture," submitted for publication, 1991.

D. Prieto-Diaz, "Domain Analysis for Reuseability," Proc. Computer Software and Applications Conf. (COMPSAC'87), Tokyo, Japan, pp. 23-29, October 1987.

D. Prieto-Diaz, "Domain Analysis: An Introduction," ACM SIGSOFT Software Engineering Notes, 15(2), pp. 47-54, April 1990.

S. Shlaer, S. Mellor, "An Object-Oriented Approach to Domain Analysis," ACM Software Engineering Notes, 14(5), pp. 66-77, July 1989

S. Wartik, R. Prieto-Diaz, "Criteria for Comparing Domain Analysis Approaches," Int. J. Software Engineering and Knowledge Engineering, 1992.

J. Webb, "Steps Toward Architecture-Independent Image Processing," Computer, IEEE, pp. 21-31, February 1992.

5. Information Infrastructure and Data Base

M. McGrath, "F-22 IWSDB Workshop II - Architectural Considerations," F-22 IWSDD Workshop II, DARPA, June 1, 1992.

J. Bauer, "EDIF Demystified," Los Altos, CA 94022, ASIC & ECA, July 1992.

*J. Gerdeen, "Engineering Data Management System," Honeywell, January 1992.

*J. Gerdeen, "Product Data Management System," Honeywell, February 1992.

D. Hughes, "Computer Infrastructure Critical to Success In Aerospace Industry," Aviation Week and Space Technology, June 22, 1992.

V. Hunt, "Enterprise Integration Sourcebook: The Integration of CALS, CE, T M, PDES, RAMP, and CIM," San Diego, CA 92101: Academic Press, Inc., 1991.

"Industry Seeks Interface Standards to Gain Full Benefit of CAD/CAM," Aviation Week and Space Technology, June 22, 1992.

"The Latest on CAD Frameworks," Electronic Design, August 6, 1992.

P. McGill, Sales Brochure, Team One Systems, Inc.

T. Scallan, "CAD Framework Initiative - A User Perspective," Kingston, NY 12402, IBM Corporation, 1992. *Note: 1992 Design Automation Conference, Session 40.2.*

J. Templer, "Component Information Systems for Electronics Design and Manufacturing Sales Brochure," Waltham, MA 02154, Aspect Development, Inc, 1992.

6. Test and Evaluation

S. Natarajan, B. Herman, "Comparison of Testability Analysis Tools for USAF," Final Report to AFOSR, February 1991. *Analyzes and compares several testability analysis tools available on the market.*

M. Abadir, "Multichip Systems Test Methodology Report," MCC Technical Report, 1991. *Reviews intelligent CAT tools for designing MCM in the areas of testability advise and guidance, test synthesis, test pattern generation, and test pattern grading.*

M Abadir, "Testability Insertion Guidance Expert System (TIGER)," MCC Technical Report, 1989. *Provides an overview of a knowledge-based system which provides designers with a systematic approach for creating complex, testable designs.*

R. Hartley, et al., "A Rapid-Prototyping Environment for Digital-Signal Processors," EEE Design & Test of Computers, pp. 11-26, June 1991. *Describes GE's Diodes environment for designing DSP.*

D. Karpenske, C. Talbot, "Testing and Diagnosis of Multichip Modules," Solid State Technology, pp. 24-26, June 1991. *Describes how boundary scan and electron beam probing can be used to test and diagnose MCM.*

R. Wagner, J. Hagge, "Improving MCM Assembly Yields through Approaches for Known-Good ICs," Rockwell International Corporation. *Discusses the problems, issues and advances needed to bring "Known-Good ICs" to the MCM assembly process.*

P. Nagvajara, M. Karpovsky, L. Levitin, "Pseudorandom Testing for Boundary-Scan Design with Built-In Self-Test," IEEE Design & Test of Computer, pp. 58-65, September 1991. *Describes the design of a pseudorandom pattern generator for a boundary-scan chip with built-in self-test.*

D. Bhatt, T. Steeves, D. Lee, "A Tool for Real-Time Performance Monitoring of Parallel and Distributed Systems," Proceedings of IEEE TENCON'92, August 1991. *Describes the concepts, architecture, and features of a tool for instrumenting parallel and distributed systems.*

ABET Technical Advisory Group, "Report on the Study of User Needs and Activities Model for ABET," Final Draft, May 1992. *Provides an analysis of the information needs and activities of potential ABET users.*

M. Abadir, et al., "Partitioning Hierarchical Designs For Testability," MCC Technical Report, 1991. *Describes the partitioning subsystem of MCC's Testability Insertion Guidance Expert System (TIGER).*

"Printed Wiring Assembly Design For Testability—Test Strategy Development and Implementation Guidelines," Honeywell CAD/CAM Report, 1990. *Provides a structured format for use by design and test engineers in the cooperative development of PWA test strategies and the application of guidelines.*

IEEE 1149.1, IEEE Standard Test Access Port and Boundary-Scan Architecture, IEEE Computer Society, 1989. *Describes the IEEE 1149.1 standards for boundary-scan.*

Military Standard 1814 - Integrated Diagnostics, June 1991.

Military Standard 2165 - Testability Program for Electronic Systems and Equipments.

7. Manufacturing, Design/Manufacturing Interface, and Cost

G. Hays, D. Wineberg, "Commilitary-Another New World Order," IEEE AES Systems Magazine, p. 16, September '92. *Past selection, environmental conditions, levels of inspection and test, configuration control of commercial parts and practices applied to military systems (modular avionics radar).*

"21st Century Manufacturing Enterprise Strategy," Iacocca Institute, Lehigh University, November 1991. *Competitive advantage will belong to agile manufacturing enterprises capable of responding rapidly to demand for high quality, high customized products.*

"Chip Aligner Bonders," Research Devices, Picataway, NJ 08854, 1992. *Product description of a prototype for automatic MCM assembling.*

J. Biancini, "Implementing Concurrent Engineering for Surface Mount Technology," Electronic Packaging & Production, August 1992. *Supplement Concurrent Engineering and Design for Manufacturability are Integral to a Successful SMT Manufacturing Operation.*

Manufacturing 21 Report, "The Future of Japanese Manufacturing," Wheeling, IL 60090, Association for Manufacturing Excellence.

CACE/DMS Sales Brochure - Perceptronics.

CACE/PM Sales Brochure - Perceptronics.

D. Carter, B. Baker, "Concurrent Engineering: The Product Development Environment for the 1990's," Addison-Wesley Publishing Company, 1991, ISBN: 0-201-56349-5.

D. Smith and Associates, Inc. Manufacturing Resource Planning, 1987.

V. Hunt, "Enterprise Integration Sourcebook: The Integration of CALS, CE, T M, PDES, RAMP, and CIM," San Diego, CA 92101: Academic Press, Inc., 1991.

*T. Jenne, J. Hunger, P. Zumsteg, A. Anderson, K. Mikkilineni, "1990 Manufacturing/ Engineering Architecture Project Final Report," Minneapolis, MN 55418: Honeywell, 30 January 1991. *Note: Volume 1: product design process model.*

*T. Jenne, J. Hunger, P. Zumsteg, A. Anderson, K. Mikkilineni, "1990 Manufacturing/Engineering Architecture Project Final Report," Minneapolis, MN 55418: Honeywell, 30 January 1991. *Note: Volume 2: appendices to the product design process model.*

L. Litton, D. Hall, K. Hutchinson, D. Hoffman, S. Evanczuk, P. Sullivan, "First Principles of Concurrent Engineering: A Competitive Strategy for Electronic Product Development," Springfield, VA: CALS/Concurrent Engineering Task Group—Electronics Systems, 30 September 1991. *Note: Reproduced by U.S. Department of Commerce, National Technical Information Service. Draft Copy CALS Technical Report 005.*

N. McEachron, R. Tara, "Reducing Time to Market: Selecting and Executing Accelerated Development Strategies," SRI International, March 1990.

L. Menker, "Results of the Aeronautical Systems Division Critical Process Team on Integrated Product Development," Wright-Patterson AFB, OH 45433-6503, November 1990.

J. Pennel, M. Slusarczuk, "An Annotated Reading List for Concurrent Engineering," Alexandria, VA, Institute for Defense Analyses, July 1989. *Note: IDA Document D-571. Documents acquired by Defense Technical Information Center (DTIC).*

Sales Brochure, "Interactive Development Environments."

S. Shina, R. Redy, R. Wood, K. Cleetus, J. Turino, "Concurrent Engineering," New York, NY 10017, IEEE Spectrum, Vol. 28, No. 7, July 1991.

R. Wilson, "Military Goes Commercial and Vice Versa," Manhasset, NY, Electronic Engineering Times, August 17, 1992.

R. Winner, J. Pennelee, H. Bertrand, M. Slusarczuk, "The Role of Concurrent Engineering in Weapons System Acquisition," Alexandria, VA: Institute for Defense Analyses, December 1988. *Note: IDA Report R-338.*

R. Winner, et al., "The Role of Concurrent Engineering in Weapons System Acquisition," IDA Report R-338, Institute for Defense Analyses, Alexandria, VA (available through NTIS), 1988.

R. Sprague, K. Singh, "Concurrent Engineering in Product Development," Design and Test of Computer, IEEE, March 1991.

C. Kelly, et al., "Findings of the U.S. Department of Defense Technology Assessment Team of Japanese Manufacturing Technology," Tech Report CSDL-R-2161, US DoD, Washington, DC, June 1989.

A. Rosenblatt, G. Watson, "Special Report on Current Engineering," Spectrum, IEEE, July 1991.

8. Voice of Customer and RASSP Cross-functional Team

B. Smith, "The Flo[illegible]heet: Animation Used to Analyze and Present Information About Complex Systems," [illegible]DPMA Virtual Reality Conference, Arlington, VA, June 1-2, 1992. *Visualization tool for past data and future projections of complex system behavior.*

"21st Century Manufacturing Enterprise Strategy," Iacocca Institute, Lehigh University, November 1991. *Competitive advantage will belong to agile manufacturing enterprises capable of responding rapidly to demand for high quality, high customized products.*

Manufacturing 21 Report, "The Future of Japanese Manufacturing," Wheeling, IL 60090, Association for Manufacturing Excellence.

Data Integration Solutions for Electronic Design Sales Brochure - DataXpress.

S. Galatowitch, "DESC Means Mil-Spec, Mil-Std," Englewood, CO 80111: Defense Electronics, Vol. 24, No. 7, July 1992. *Electronic components that meet rigid military specifications are mandated for use in many systems, but by what process are they militarized, how is reliability guaranteed, and how do they end up in the hands of a system integrator?*

V. Hunt, "Enterprise Integration Sourcebook: The Integration of CALS, CE, T M, PDES, RAMP, and CIM," San Diego, CA 92101: Academic Press, Inc., 1991.

B. King, "Better Designs in Half the Time: Implementing QFD Quality Function Deployment in America," Methuen, MA 01844, GOAL/QPC, 1989. *Note: third edition*

L. Litton, D. Hall, K. Hutchinson, D. Hoffman, S. Evanczuk, P. Sullivan, "First Principles of Concurrent Engineering: A Competitive Strategy for Electronic Product Development," Springfield, VA: CALS/Concurrent Engineering Task Group—Electronics Systems, 30 September 1991. *Note: Reproduced by U.S. Department of Commerce, National Technical Information Service. Draft Copy CALS Technical Report 005.*

N. McEachron, R. Aldrige Tara, "Reducing Time to Market: Selecting and Executing Accelerated Development Strategies," SRI International, March 1990.

R. Winner, J. Pennelee, H. Bertrand, M. Slusarczuk, "The Role of Concurrent Engineering in Weapons System Acquisition," Alexandria, VA: Institute for Defense Analyses, December 1988. *Note: IDA Report R-338.*

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 12 October 1992	3. REPORT TYPE AND DATES COVERED Final Report for Period 5/12/92 - 10/12/92

4. TITLE AND SUBTITLE Rapid Prototyping of Application Specific Signal Processors (RASSP) Program - Study Phase	5. FUNDING NUMBERS
6. AUTHOR(S) Fred Malver, Andrzej Peczalski, Wing Au, Jonathon Krueger, David Lee	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Honeywell Inc. Systems and Research Center 10701 Lyndale Avenue South Bloomington, MN 55420	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency Electronic Systems Technology Office 3701 North Fairfax Drive Arlington, VA 22203-1714	10. SPONSORING/MONITORING AGENCY REPORT NUMBER MDA972-92-C-0057
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION/AVAILABILITY STATEMENT Unrestricted	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This is the final report for the Rapid Prototyping of Applications Specific Signal Processors (RASSP) Program - Study Phase. This study represents a five-month contract effort, DARPA contract number MDA972-92-C-0057, performed by Honeywell's Systems and Research Center for the Defense Advanced Research Projects Agency, DARPA.

The broad objective of this study was to produce information that would aid the Government program manager to manage the risks inherent in the implementation phase of the RASSP program. This meant (1) identifying the risks and problem areas that might be encountered during the implementation phase and (2) suggesting risk reducers or solutions that could be used to minimize or solve these problems. Ultimately this meant recommending an approach for the implementation phase and also identifying potential areas for further work.

The RASSP program embraces two tightly coupled focuses; one related to process or methodology, and the other related to target prototype or product systems. These are discussed in detail in this final report.

14. SUBJECT TERMS Rapid Prototyping, Application Specific Signal Processors, Design, Manufacturing and Field Support Process and Methodology Infrastructure, Model Year Upgrades, Target Applications - ATR, EW, Communications, and SIGINT.	15. NUMBER OF PAGES 16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Same as report